Pascal Gachet
Travail de diplôme 2001

Déploiement de solutions VPN : **P**KI Etude de cas

Département E+I
Filière : **Télécommunication**
Orientation : **Réseaux et services**
Professeur responsable : **Stefano Ventura**
Date : **20 décembre 2001**

Remerciements

Dans le cadre de ce travail de diplôme, je tiens à remercier sincèrement.
.
M. S.Ventura pour son soutien et sa disponibilité, qui m'ont permis de progresser et d'évoluer dans ce projet.

M. S. Maret, qui par ses nombreux conseils, a su me donner une ligne directrice avisée d'un œil d'expert.

M.C.Tettamanti pour son expérience pratique sur les VPN et sa connaissance du système LINUX.

M.F.Bucher pour son aide précieuse dans le domaine de LINUX.

Avant-Propos

A l'heure de la nouvelle économie, l'utilisation du protocole Internet (IP) comme base des réseaux informatiques est devenue une tendance tout à fait marquante.
Ce protocole est né d'un besoin naturel d'échanger des informations de manière simple et souple dans un cadre de travail informatique. Mais l'avènement d'Internet a également modifié considérablement nos façons de communiquer, de travailler, et de consommer.

La possibilité d'être constamment en contact informatique avec une masse d'utilisateurs globale et hétérogène a permis d'entrevoir de nouvelles possibilités pour gérer notre quotidien, e-mail, e-commerce etc

Ces nouvelles technologies ont également entraînés dans leur ascension de nouveaux problèmes qui sont liés à la sécurité informatique, hacker ; propagation de virus en sont des exemples.

Le protocole Internet (IP) n'a jamais été prévu pour garantir la sécurité des transactions. Ce besoin évident de sécurité a engendré le développement de diverses solutions de sécurité comme les firewall, routeurs filtrants, etc.

Mais ces différents mécanismes ne permettent pas de déployer une solution parfaitement sécurisée pour Internet, car des grandes questions restent en suspend.

1) Mes données ont-elles été modifiées en route ?
2) La personne avec laquelle je communique est-elle bien celle qu'elle prétend être ?
3) Peut-on épier mes conversations ?

Tant que ces questions ne sont pas résolues, Internet ne peut pas être un média assez sûr en soi pour déployer des applications de e-commerce, de télétravail ou télébanking.

Dès lors, les connexions sécurisées doivent reposer sur des médias propriétaires, ce qui représente un investissement considérable pour les entreprises désireuses d'acquérir ces nouvelles technologies dans leur panel de fonctionnalité.

Devant les besoins grandissants dans le domaine de la sécurité informatique, et profitant de la définition du nouveau protocole Internet Ipv6, l'IAB (Internet Architecture Board) a donc décidé d'intégrer des services de sécurité dans le protocole IP lui-même, afin de pouvoir protéger les communications utilisant ce protocole.
Mais le réseau Ipv4 étant encore largement déployé et la migration complète vers Ipv6 nécessitant encore beaucoup de temps, il est vite apparu intéressant de définir des mécanismes de sécurité qui soient communs à la fois à Ipv4 et Ipv6.

L'idéal serait d'exploiter le protocole IP et sa souplesse tout en ajoutant une couche sécurisée supplémentaire absolument nécessaire aux applications modernes comme le e-commerce, VPN, e-banking, e-voting, ERP.

Objectifs

Dans le cadre de ce travail de diplôme, les forces ont été concentrées sur le besoin de sécurité propre au VPN (Virtual Private Network).

Les réseaux VPN ont pour but de permettre une connexion sécurisée à travers Internet, depuis un poste quelconque jusqu'à une passerelle VPN appelée gateway.
Le but final étant d'accéder à son réseau local d'entreprise (LAN Local Area Network) par l'intermédiaire d'un réseau public comme Internet. (figure1)
Une connexion VPN peut utiliser différents mécanismes pour aboutir à une connexion sécurisée, notamment le concept «d'encapsulation chiffrée ». Ce mécanisme permet de créer au niveau du réseau un tunnel virtuel. Le tunnel VPN protège le flux de données par des mécanismes puissants de cryptographie qui n'ont pas présenté de faiblesse connue jusqu'ici.

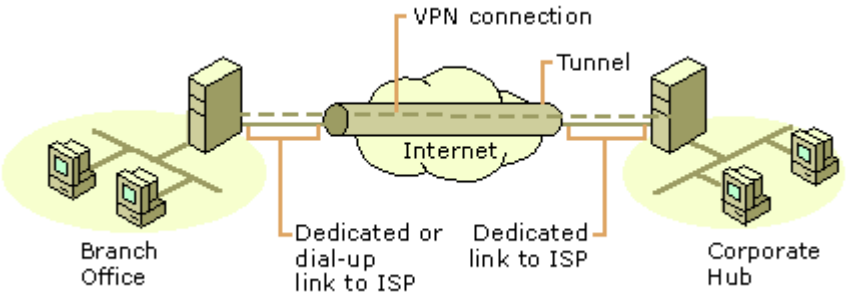


Figure 1 *Connexion VPN*

D'un point de vue commercial, les VPN sont extrêmement intéressants, car ils reposent sur le réseau Internet existant et largement déployé, diminuant de manière très sensible le coût qui aurait été engendré par l'utilisation de ligne louée.

Cette possibilité a poussé le CCTI (centre de compétence de HES-SO dans les technologies de l'information) à financer un projet VPN dans le cadre des laboratoires de l'EIVD et de l'EIG. Ce projet permettra de simuler la problématique d'une entreprise répartie sur deux sites géographiquement séparés, qui doit partager des zones de ressource.

Le projet permettra également à des utilisateurs distants, de se connecter depuis n'importe quel poste au réseau d'entreprise. Simulant ainsi le cas du télétravail.

Problématique

Si le VPN permet de protéger le flux de données par une encapsulation cryptée, cette encapsulation ne suffit pas à combler tous les besoins de sécurité.
Avant d'établir un tunnel VPN, et ainsi de protéger les données, les deux interlocuteurs ont besoin de s'authentifier mutuellement.

L'authentification est indispensable à tout connexion sécurisée !

Le protocole Internet IP ne fournit pas les outils nécessaires à une authentification fiable. Internet utilise une politique d'adressage qui n'est pas suffisante pour garantir l'identité des interlocuteurs.

Sur Internet l'utilisateur est considéré comme anonyme !

Le travail qui me revient dans le projet VPN consiste à étudier, définir, à appliquer un moyen d'authentification fort et efficace dans le cas précis d'un VPN.
La solution doit être adaptée au cas précis du VPN, mais elle devrait également être polyvalente et s'appliquer à d'autres applications modernes nécessitant une authentification.

Actuellement, le standard d'authentification pour Internet se base sur le principe de certificat numérique. Un certificat numérique est comme un passeport, il contient une preuve d'identité crédible et irréfutable. Comme toute pièce d'identité, le certificat numérique doit être délivré par un organisme de confiance reconnu par tous les utilisateurs.

Il existe un grand nombre d'autorités capables de fournir de tels certificats sur le marché, ces autorités portent le nom générique de PKI (Public Key Infrastructure). Leur capacité à générer des pièces d'identité numériques est un service qui n'est pas gratuit. Un certificat numérique, comme toute pièce d'identité est sujette à un coût qui peut être relativement élevé suivant le service fourni par la PKI.

Les besoins d'authentification pour le cas spécifique d'un VPN nécessitent une grande quantité de certificats, chaque utilisateur doit disposer d'un certificat personnel.

Pour cette raison, il est nécessaire dans ce projet, de disposer de notre propre réseau de distribution de certificats numériques, c'est-à-dire de notre propre PKI.

La politique suisse en vigueur autorise la mise en œuvre d'une autorité de certification PKI privée.

Ce travail de diplôme aura donc pour but de concevoir une autorité de certification propre à l'EIVD, cette autorité devra engendrer le minimum de coût possible, pour cette raison l'univers opensource fourni par LINUX est requis.

La réalisation finale consistera à combiner le mécanisme puissant d'authentification fourni par la PKI au chiffrage efficace mis en œuvre dans le cadre d'un VPN.

Décomposition du travail

Le projet VPN est en réalité un vaste projet qui a été décomposé en plusieurs projets de diplôme dans plusieurs écoles d'ingénieur, notamment le travail effectué par C.Tettamanti dans le cadre de l'EIVD (banc de teste VPN) diplôme 2000.

- Le travail de semestre visait à reprendre le travail effectué par C.Tettamanti pour s'initier à la problématique des VPN. Durant cette période, les différentes solutions VPN ont pu être étudiées et analysées par la lecture du tutorial et la réalisation du laboratoire VPN.

  Dans son travail, de nombreuses solutions VPN ont été étudiées, C.Tettamanti a configuré et testé des connexions VPN basées sur plusieurs protocoles comme Ipsec, PPTP.
  Il a également testé des implémentations clientes comme SafeNet, Freeswan et WIN2000.

  Si il a su mettre en évidence la possibilité de mettre en pratique une solution VPN opensource dans le cadre de l'école, il a également soulevé le problème crucial de l'authentification qui n'était pas gérée de manière satisfaisante.

  Il n'était pas non plus possible avec sa solution de permettre à un nombres important de client de se connecter au VPN depuis des postes quelconque (client nomade ou road warrior).

- Il s'agissait donc d'étudier les différents mécanismes d'échange des clés et d'authentification des différents protocoles VPN. L'étude s'est portée de manière spécifique au protocole `IPsec`, le résultat est publié dans le document « Gestion des clés pour Ipsec ».
  La possibilité d'utiliser des certificats numériques dans le cadre d'Ipsec est apparue dans cette étude, elle à permis d'entrevoir une solution basée sur une PKI.

- Le travail de diplôme a débuté par une étude théorique des différents mécanismes de cryptographie rencontrés dans les applications sécurisées. Cette étude est contenue dans le document « sécurité et PKI ». Il contient la théorie qui a permis de déployer une solution Pki, mais également une partie théorique sur d'autres systèmes d'authentification alternatifs à la PKI. Les points étudiés sont les suivants :

  1. Chiffrage symétrique asymétrique.
  2. Signature numérique.
  3. Echanges des clés.
  4. Authentification Kerberos.
  5. Authentification PKI.
  6. Composants et mécanismes PKI

  Le document intitulé « *sécurité et PKI* » a été rédigé dans le but de constituer une référence théorique sur l'ensemble du travail de diplôme. Mais son but est également d'être un document à des fins pédagogiques. A cet effet, il comporte une série de questions.

En plus de constituer une partie intégrante de ce mémoire, il est également disponible de manière indépendante en annexe sur CD, sous forme de tutorial.

- Le travail s'est poursuivi par une analyse d'un produit PKI commercial, il s'agit du produit Keon développé par RSAsecurity.
  L'étude de ce produit a permis en premier lieu de s'initier de manière pratique à un outil de travail PKI et ainsi de faire la liaison entre les mécanismes théoriques et la réalité.
  Cette étude a permis de définir une liste des différentes fonctionnalités offertes par une solution commerciale. Le but avoué étant de constituer un cahier des charges des fonctionnalités indispensable à retrouver dans une solution logiciel PKI openSource.

- La mise en œuvre d'une PKI Opensource a ensuite été déployée en suivant toutes les contraintes de sécurité nécessaires à sa mise en œuvre dans un environnement grandeur nature.
  La solution s'est basée sur OpenCA.
  Elle permet de fournir les fonctionnalités requises pour une intégration avec un VPN.

  Pour permettre un développement futur de cette solution, un document précis a été rédigé. Il contient non seulement les différentes étapes qui ont permis la mise en œuvre de la PKI, mais aussi une motivation précise des choix effectués durant cette phase d'intégration. Pour cette raison, ce document fait partie intégrante du mémoire (12 Mise en œuvre d'une PKI Open Source pour Linux).
  Ce document contient toutes les étapes permettant d'installer une PKI basée sur OpenCA, mais la manipulation de la PKI n'est pas explicitée dans ce document, la partie manipulation qui correspond au « manuel d'utilisation » est contenue dans un document de laboratoire. (15 Laboratoire PKI)

  La partie mise en œuvre de la PKI Opensource a représenté la phase la plus importante et la plus longue du travail de diplôme.

- Un laboratoire spécifique à la technologie PKI a ensuite été réalisé, il devra s'intégrer au document « sécurité et PKI » pour fournir un support didactique et pratique destiné aux futur étudiants.
  Le laboratoire permet d'apporter aux utilisateurs une compréhension des mécanismes PKI basée sur la manipulation des différents éléments PKI, depuis la sollicitation d'un certificat jusqu'à l'obtention de celui-ci, en passant par toutes les phases de contrôle, signature et distribution.
  Le laboratoire s'est axé sur la problématique Pki, les aspects publications par LDAP ne sont volontairement pas abordés dans ce laboratoire pour éviter une confusion des utilisateurs.
  La problématique LDAP est un vaste sujet à part entière qu'il est indispensable d'étudier, en premier lieu dans un contexte différent, avec un laboratoire spécifique.

  Le laboratoire est également constitué de différentes manipulations qui permettent de comprendre et d'apprécier le mécanisme d'authentification apporté par les certificats numériques dans le cadre du protocole SSL.

- La phase finale du travail a consisté à tester l'interopérabilité de la technologie PKI apportée par les certificats numériques au VPN basé sur Ipsec.
  Cette phase a permis de résoudre les problèmes d'authentification de façon satisfaisante. Elle a également permis de résoudre le problème des utilisateurs nomades (road warrior).

- Une étude théorique a ensuite été entreprise pour permettre de définir des politiques d'accès des différents groupes d'utilisateurs pouvant se connecter au VPN. Par exemple, un professeur devrait avoir des privilèges différent de ceux d'un étudiant sur le service fourni.
  Cette étude s'est basée sur différentes possibilités de classer et de séparer les utilisateurs. Les différentes possibilités d'extension des certificats numériques ont été abordées.

Composition du mémoire

Le mémoire, tel qu'il est présenté dans ce document ne suit pas l'ordre chronologique défini dans le cahier des charges. Le déroulement de ce document suit une voie logique pour permettre une compréhension incrémentale du travail dans son ensemble.

Il est composé

- Du tutorial « PKI et sécurité »
- De l'étude du produit Keon
- De la mise en œuvre d'une PKI opensource
- Du laboratoire PKI.
- Des mécanismes d'intégration des service PKI dans le cadre d'un VPN
- De l'étude des différentes variantes de classification des utilisateurs.

En annexe :
- Sigles et acronyme
- Du document « gestion des clés pour Ipsec »

Table des matières

Remerciements *2*
Avant Propos *3*
Objectifs *4*
Problématique *5*
Décomposition du travail *6*
Composition du mémoire *8*
Table des matières *9*
Table des illustrations *15*
1. Cryptographie *18*
1.1 rôle de la cryptographie 18
2. cryptographie à clés symétriques et asymétriques *19*
2.1 Algorithmique à clés symétriques 19
2.1.1 Algorithmes de chiffrement par blocs 19
2.1.2 Mode ECB 20
2.1.3 Mode CBC 20
2.1.4 Mode CFB 21
2.1.5 DES 21
2.2 Algorithmes à clés asymétriques 22
2.2.1 Fonction à sens unique 22
2.2.2 Fonction de hachage à sens unique 23
2.2.3 Limitation de la cryptographie à clé publique. 24
2.2.4 RSA exemple d'algorithme à clé asymétrique. 25
2.3 Échange de clés à l'aide de la cryptographie à clé publique 26
2.4 échange des clés par Diffie –hellmann 28
3 Authentification. *30*
3.1 But de l'authentification 30
3.2 Authentification asymétrique 30
3.2.1 Signature numérique. 30
3.2.2 Signature par la clé privée. 30
3.2.3 Signature par fonction de hachage et clé publique 31
3.3 Authentification symétrique 32
3.3.1 Authentification par scellement. 32
4 Échange de clés et authentification *33*

4.1 Définition des clés 33
4.2 Propriété des protocoles d'échange de clés. 33
5 Authentification à l'aide d'une tierce personne de confiance *35*
5.1 Signature de documents à l'aide d'un cryptosystéme à clé symétrique et d'un arbitre 35
5.2 Kerberos 35
5.2.1 Fonctionnement de Kerberos 35
5.2.2 Description générale Kerberos version 5 36
5.2.3 Description détaillée 37
5.2.4 Sécurité de Kerberos 38
6 Public Key Infrastructure *39*
6.1 Besoin d'un organisme de gestion des clés 39
6.2 PKI définition 39
6.3 Environnement sécurisé 41
6.3.1 Classification des ressources 41
6.3.2 Séparer les zones publiques des zone privées 41
6.3.3 Protection contre les accidents 41
6.4 Gestion des clés 42
6.4.1 Génération des clés 42
6.4.2 Distribution des clés 43
6.4.3 Stockage des clés 43
6.4.4 Suppression de clés 43
6.4.5 Archivage des clés 43
6.4.6 Recouvrement des clés (Key Recovery) 44
6.5 Composant d'une PKI 44
6.5.1 Autorité d'enregistrement (RA) 44
6.5.2 Autorité de certification (CA) 45
6.5.3 Application compatible PKI (PKI-ready) 46
6.6 Répartition des CA 47
6.6.1 Modèle hiérarchique 47
6.6.2 Modèle Peer to peer 48
6.6.3 Modèle en pont 49
6.7 Politique de certification 49
6.8 Le certificat X509 50
6.9 Service de révocation CRL 52
6.10 Service de publication 52
7 Annuaire *54*

7.1 Annuaire et PKI 54
7.2 Annuaire définition 54
7.3 Rôle de l'annuaire dans une PKI 55
7.4 Protocole d'accès au répertoire 55
7.4.1 X.500 56
7.4.2 LDAP 56
7.4.3 Architecture LDAP 57
7.4.4 Sécurité d'accès à l'annuaire 58
8 Protection des clés privées 59
8.1 accès aux clés privées 59
8.2 Smart Cards 59
9 Politique de sécurité 60
9.1 Références légales 60
9.1.1 Rapport pratique de certification (CPS) 60
9.1.2 Politique de certificat 61
9.1.3 Considération légal 61
10 PKI et authentification bio métrique 61
10.1 bio métrie définition 61
10.2 Choix d'une technique bio métrique dans le cadre d'une PKI 62
Conclusion 63
Questions de révisions 64
Bibliographie 66
11 Étude d'une PKI commerciale 68
11.1 Généralités 68
11.2 But de l'étude 68
11.3 Installation 68
11.4 Architecture PKI de KEON 69
11.4.1 Serveur d'administration (Administration Server) 69
11.4.2 Serveur d'enrôlement (Enrollement Server) 70
11.4.3 Serveur des répertoires sécurisés (Secure Directory Server) 70
11.4.4 Logging server 70
11.5 Architecture CA 71
11.6 Service de révocation 72
11.7 Procédure d'enrôlement 72
11.8 Key recovery module 73
11.9 Credential store 73

Conclusion ... *73*
Références ... *74*
12 Mise en œuvre d'une PKI open source pour Linux ... *75*
12.1 Introduction ... 75
12.2 Choix d'un projet pour une solution Open PKI ... 75
12.3 Architecture open PKI ... 77
12.3.1 Serveur Public ... 78
12.3.2 Serveur RA ... 78
12.3.4 Serveur CA ... 79
12.4 Rôles des membres PKI ... 79
12.4.1 Rôles des clients ... 79
12.4.2 Rôle de l'administrateur de la RA ... 79
12.4.3 Rôle de l'administrateur de la CA ... 80
12.5 Zone d'enrôlement ... 80
12.6 Hiérarchie de CA ... 81
12.7 Protection des clés privées ... 81
12.8 Key recovery ... 82
12.9 Liste de révocation ... 82
12.10 Interopérabilité ... 82
13 PKI open source avec OpenCA ... *83*
13.1 Projet OpenCa ... 83
13.2 Préliminaire pour OpenCa 0.8.0 ... 84
13.2.1 OpenSSL ... 84
13.2.2 Installation d'un interpréteur Perl. ... 85
13.2.3 Installation script perl et modules spécifiques ... 85
13.2.4 Installation OpenLdap sur le poste de la RA ... 86
13.3 Installation de OpenCa 0.8.0 (Partie CA) ... 86
13.3.1 Pré configuration OpenCa ... 86
13.3.2 Installation de la CA ... 87
13.4 Installation de OpenCa 0.8.0 (Partie RA et interface publique) ... 89
13.4.1 Pré configuration OpenCa ... 89
13.4.2 Installation Ra et interface publique ... 90
13.5 Apache ... 91
13.5.1 mode SSL ... 92
13.5.2 Configuration du serveur apache ... 92
13.5.3 Configuration serveur de la CA ... 93

13.5.4 Initialisation de la CA 94
13.5.5 Configuration serveur de la RA 96
13.5.6 Droit d'accès au serveur RA 103
13.5.7 Configuration serveur public 106
14 LDAP *111*
14.1 configuration serveur LDAP 111
14.2 Hachage du mot de passe manager 113
14.3 Contrôle des droits d'accès 113
14.4 Initialisation de l'annuaire LDAP 114
14.4.1 Initialisation par un fichier LDIF 115
14.4.2 Initialisation automatique. 117
14.5 Client LDAP 118
15 Laboratoire PKI *121*
15.1 Description de la maquette PKI 121
15.1.1 Serveur Public 122
15.1.2 Serveur RA 122
15.1.3 Serveur CA 122
15.2 Rôles des membres PKI 123
15.2.1 Rôles des clients 123
15.2.2 Rôle de l'administrateur de la RA 123
15.2.3 Rôle de l'administrateur de la CA 123
15.3 Zone d'enrôlement 124
15 .4 Manipulation de la PKI 124
15.4.1 Obtention du certificat CA root 124
15.4.2 Sollicitation d'un certificat 126
15.4.3 Administration de la RA 127
Exporter le certificat client 130
15.4.4 Administration de la CA 131
15.4.5 Réception du certificat 132
15.4.6 Liste de révocation 133
15.5 Etudes d'échange de certificat dans le cadre du protocole SSL 138
15.5.1 Etude du protocole SSL 138
15.5.2 Connexion SSL sans authentification client. 139
15.5.3 Connexion SSL avec authentification client 140
16 Intégration de la technologie PKI dans le cadre d'une solution VPN basée sur Ipsec *142*
16.1 Introduction 142

16.2 Analyse d'échange des clés et d'authentification pour Ipsec 143
16.2.1 Echange avec protection de l'identité (identity Protection) 144
16.3 Analyse de différent mécanisme d'échange des clés pour IPsec 145
16.3.1 Configuration Host to Lan avec PSK 145
16.3.2 Authentification par signature RSA 146
16.3.3 Authentification par signature RSA et certificat numérique 148
16.3.4 Authentification par échange de certificat dans un bloc ISAKMP 152
16.4 Contrôle des certificats 153
16.4.1 Contrôle de la signature de la CA 154
16.4.2 Contrôle de la liste de révocation CRL 154
16.5 Configuration en road warrior 154
17 Méthode de classification des groupes utilisateurs *155*
17.1 Motivation 155
17.2 Classification par mot de passe et login 155
17.3 Définition des extensions x509v3 156
17.4 Définition des groupes d'utilisateurs par les extension V3 160
17.5 Classement par signature de la CA 161
17.6 Classement d'après le DN du certificat 162
17.7 Contrôle d'accès centralisé 163
Conclusion *165*
Références *166*
18 Conclusion générale *167*
Annexe *168*
Annexe A Sigles et acronyme 168

Table des illustrations

Figure 1 *Connexion VPN* **4**
Figure 2 *clé symétrique* **19**
Figure 3 *clés asymétriques* **22**
Figure 4 *fonction de hachage* **24**
Figure 5 *Chiffrage hybride* **27**
Figure 6 *Diffie-Hellmann* **29**
Figure 7 *Signature numérique* **31**
Figure 8 *Scellement* **32**
Figure 9 *Kerberos* **36**
Figure 10 *PKI* **40**
Figure 11 *Multi CA* **47**
Figure 12 *Ca root* **47**
Figure 13 *Co-certification* **48**
Figure 14 *CA Bridge* **49**
Figure 15 *certificat X509* **51**
Figure 16 *Hiérarchie LDAP* **57**
Figure 17 *Architecture KEON* **69**
Figure 18 *Hiérarchie CA* **71**
Figure 19 *Peer To Peer CA* **71**
Figure 20 *Architecture open PKI* **77**
Configuration 1 *Module Perl de la CA* **87**
Configuration 2 *Lien sur Openssl (extrait ca.conf)* **88**
Configuration 3 *Définition des groupes d'utilisateurs (extrait_public.conf)* **90**
Configuration 4 *Interface RA/CA (extrait raserver.conf)* **91**
Configuration 5 *Droit d'accès sur les répertoires de la CA (extrait commonhttpd.conf)* **94**
Configuration 6 *Interface CA/RA (extrait ca.conf)* **96**
Configuration 7 *Virtual host serveur RA(extrait raSSL.conf)* **98**
Configuration 8 *Paramètres SSL serveur RA (extrait raSSL.conf)* **98**
Configuration 10 *Ajout de la configuration raSSL.conf (extrait httpd.conf)* **99**
Figure 21 *Import du certificat administrateur* **102**
Configuration 11*Configuration avec certificat PKI (extrait raSSL.conf)* **103**
Configuration 12 *Limitation du droit d'accés par l'attribut OU (extrait raSSL.conf)* **104**
Configuration 13 *droit d'accès au répertoires de la RA (extrait .htaccess)* **105**
Figure 22 *Login administrateur* **106**
Configuration 14 *Virtual host serveur Public(extrait publicSSL.conf)* **107**
Configuration 16 *Paramètre serveur public (extrait publicSSL.conf)* **109**
Configuration 17 *section LDAP (extrait raserver.conf)* **112**
Configuration 18 *Configuration du serveur LDAP (extrait slapd.conf)* **112**
Configuration 19 *ACL (extrait slapd.conf)* **113**
Figure 24 *Groupe d'utilisateur* **114**
Configuration 20 *Exemple de schéma (extrait core.schema)* **115**
Configuration 21 *initialisation par fichier LDAP (extrait PKI.ldif)* **117**
Figure 25 *client PKI sur LDAP* **119**
Figure 26 *Architecture OpenPKI* **121**
Figure 27 *Réception du certificat Root* **125**
Figure 28 *Mise en garde du browser* **125**
Figure 29 *Format de requêtes* **126**
Figure 30 *Outils de sécurité* **128**
Figure 32 *Netscape CRL* **136**
Figure 33 *tcomCA CRL* **137**
Figure 34*Couche SSL* **138**
Figure 35 *Erreur de connexion SSL* **140**
Figure 36 *Notation pour les échanges ISAKMP* **144**

Table des illustrations

Figure 37 *Echange identity protection* **144**
Figure 38 *Connaissance préalable d'un PSK* **145**
Configuration 22 *Configuration GW pour PSK* **146**
Configuration 23 *Clé RSA* **147**
Configuration 24 *Configuration GW pour RSAsig* **147**
Figure 39 *Connaissance préalable des clés publique RSA* **148**
Configuration 25 *Configuration GW pour deux clients x509* **150**
Figure 40 *Connaissance préalable des certificats* **151**
Figure 41 *Echange des certificats* **152**
Configuration 26 *Configuration Gw pour un échange de certificast en ligne* **153**
Figure 42 *Classification par signature CA* **161**
Figure 43 *VPN basé sur la signature de la CA* **162**

Sécurité et **PKI**

1. Cryptographie

1.1 rôle de la cryptographie

De tout temps la question de la sécurité dans le transfert de données à été un problème envisagé avec le plus grand sérieux. Les militaires ont été pour des raisons évidentes confrontés très tôt à ce genre d'exigences ; jusqu'à très récemment le domaine public n'avait qu'un droit très limité à la sécurité des données. Mais le changement très marqué de nos moyen de communication, l'utilisation d'Internet pour des applications commerciales à relancé le problème crucial du droit à la sécurité, car de nombreux hacker pouvaient avec plus ou moins de facilité s'emparer et déchiffrer nos données. L'utilisateur devait avoir les mêmes privilèges que l'armée dans le traitement de ses données à caractère monétaire. A ce stade, il devenait presque évident que toutes les données puissent être traitées avec autant de sérieux que si il s'agissait d'argent ou de secret militaire. Une migration du know-how militaire en matière de sécurité s'est donc tout naturellement dirigée sur le réseau Internet.

L'art et la science de garder un secret est appelé cryptographie. De nos jours, ce sont les mathématiciens qui étudient la cryologie et cette science est exploitée par les informaticiens pour les applications
La cryptographie dans les applications téléinformatiques doit assurer.

- **La confidentialité**. Seul le destinataire peut connaître le contenu des messages qui lui sont transmis.
- **L'authentification**, le destinataire d'un message doit pouvoir s'assurer de son origine. Un intrus ne doit pas se faire passer pour quelqu'un d'autre.
- **L'intégrité des données,** le destinataire doit pouvoir s'assurer que le message n'a pas été modifié en chemin.
- **Le non désaveu.** Un expéditeur ne doit pas pouvoir, par la suite, nier à tort avoir envoyé un message.

Ces exigences sont vitales si l'on désire effectuer une communication sécurisée à travers un réseau informatique tel qu'Internet.
Il n'existe pas une méthode simple et sûre pour permettre de telles exigences, mais une palette de techniques permettent, en les combinant, de satisfaire ces besoins de sécurité ; mais il est clair que la sécurité absolue reste une utopie.
Pour chaque secret, il est nécessaire de déterminer quelles seraient les conséquences et les dégâts engendrés si le secret était percé ; à partir de l'analyse du cas on définit des degrés de sécurité et la complexité des algorithmes responsables de protéger ce secret.

Plus la complexité est large plus long sera le travail du hacker pour casser la protection, mais aujourd'hui l'immense quantité d'opérations nécessaires à cette tache peut être répartie en autant de sites indépendants augmentant ainsi la puissance de calcul des hacker. Si le coût nécessaire pour casser un algorithme dépasse la valeur de l'information chiffrée, alors cet algorithme peut être considéré comme sûr.

2. cryptographie à clés symétriques et asymétriques

2.1 Algorithmique à clés symétriques

Il y a deux types principaux d'algorithmes à base de clé, à clé symétrique ou à clé asymétrique. Les algorithmes à clé symétrique ou secrète sont des algorithmes où la clé de chiffrement peut être calculée à partir de la clé de déchiffrement ou vice versa. Dans la plupart des cas la clé de chiffrement et la clé de déchiffrement sont identiques. Pour de tels algorithmes, l'émetteur et le destinataire doivent se mettre d'accord sur une clé à utiliser avant d'échanger des messages chiffrés. (Figure 2)

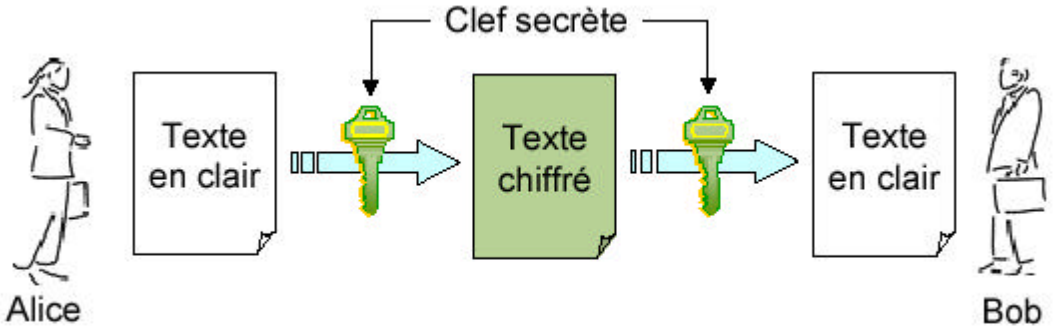


Figure 2 *clé symétrique*

Cette clé doit être gardée secrète. La sécurité d'un algorithme à clé symétrique repose sur la clé : si celle ci est dévoilée, alors n'importe qui peut chiffrer ou déchiffrer des messages.
Il existe deux types d'algorithme à clé secrète. Certains traitent le message en clair un bit à la fois, ceux ci sont appelés stream cipher pour algorithmes de chiffrement continu. D'autres opèrent sur le message en clair par groupe de bits. Ces groupes sont appelé bloc, ces algorithmes sont appelés block ciphers ou algorithme de chiffrement par bloc.

2.1.1 Algorithmes de chiffrement par blocs

Avec un tel algorithme, le même bloc de texte en clair sera toujours chiffré en un même bloc de texte chiffré, en utilisant la même clé. Ce qui n'est pas le cas pour un algorithme de chiffrement en continu, le même bit ou byte de texte en clair sera chiffré en un bit ou byte différent à chaque chiffrement. Des algorithmes comme DES, CAST et Blowfish en sont des exemples, les blocs ont une taille de 64 bits.

Pour obtenir un chiffrement par blocs il existe plusieurs méthodes, mais toutes ont en commun une sorte de rétroaction et des opérations simples

2.1.2 Mode ECB

La fonction de base pour implémenter un algorithme par block est de passer chaque bloc dans un module électronique qui chiffrera séparément les blocs pour ensuite les ré assembler, le déchiffrement se fait de la manière inverse en passant les blocs chiffrés dans des modules électroniques spécialisés qui déchiffreront les block et les rassembleront. Un tel système est appelé ECB (electronic code book), comme chaque bloc est toujours chiffré de la même manière, il est possible de définir un carnet de codage de texte en clair et de leurs textes chiffrés correspondants.

Mais pour utiliser un tel système, il est nécessaire que la taille du message à chiffrer soit la même que la taille des cellules de chiffrement, pour cela il est nécessaire d'ajouter du bourrage dans le code d'entrée, ces bits supplémentaires seront chiffrées avec le reste des données mais peuvent également être tronqués suivant l'implémentation.

Toutefois le défaut principal d'un système ECB est que si un hacker a le texte en clair et le texte chiffré de plusieurs messages, il peut commencer à construire un carnet de codes sans connaître la clé, et comme dans la réalité des fragments de code ont tendance à se répéter, comme l'entête d'une adresse IP par exemple, il pourra connaître assez d'informations pour mener des attaques contre le texte en clair sans connaître pour autant l'algorithme de chiffrement.

Mais le danger plus significatif de cet algorithme est qu'un individu mal intentionné pourrait modifier les messages chiffrés en ne connaissant pas la clé, par exemple en observant une série de messages chiffrés il s'est aperçu qu'un bloc donnait toujours le même résultat, suivant la transaction il peut découvrir le rôle de cette information et la rajouter sans autre à un autre message chiffré, le cas le plus typique est sans conteste un virement bancaire, en connaissant le résultat chiffré du compte du destinataire et le résultat chiffré de son compte personnel, il pourrait intervertir les deux informations dans le message, le hacker ne connaît pas l'algorithme, mais la forte corrélation entre les blocs clairs et les blocs chiffrés lui on permis de détourner de l'argent.

2.1.3 Mode CBC

Pour éliminer cette forte corrélation, un second système utilise une méthode de rétroaction, les résultats du chiffrement sur blocs précédents sont réutilisés comme entrées pour le chiffrement du bloc courant.

Ce qui revient à dire que le bloc chiffré ne répond pas seulement au bloc en clair, mais à tous les blocs en clair précédent. La technique de chiffrement par bloc CBC (cipher block chaining) est la suivante. Le texte en clair est combiné par X-or avec le bloc chiffré précédent avant d'être chiffré puis il servira pour le chiffrement du bloc suivant.

Le premier bloc est important, car il contient souvent des informations importantes quant à la nature du message, les entêtes des paquets par exemple, pour éviter que ce bloc ne puisse être reconnu, on combine le premier bloc avec un vecteur n'initialisation IV, ce vecteur doit être composé de valeurs aléatoires pour assurer que le résultat soit bien totalement différent de l'entrée.
De cette manière il est impossible pour un intrus de recréer un carnet de codage cohérent. De plus il peut être prouvé mathématiquement que le vecteur IV bien que devant être unique par message n'a pas besoin d'être tenu secret.
Le déchiffrement est aussi facile. Un bloc de texte chiffré est déchiffré normalement, une fois que le bloc suivant à été déchiffré, il est combiné par X-or avec le résultat du bloc précédent et ainsi de suite.

2.1.4 Mode CFB

Avec le mode CBC, le chiffrement ne peut commencer avant qu'un bloc complet de données ait été reçu. Ce défaut est particulièrement néfaste dans le cadre de réseau où un terminal doit pouvoir transmettre chaque caractère à l'ordinateur central dès qu'il est entré. En résumé, lorsque les paquets sont plus petits que 64 bits le mode CBC est à éviter.

Le mode CFB (Cipher Feed Back) quant à lui permet de chiffrer des données par unités plus petites que la taille de bloc, mais tout comme le mode CBC, le mode CFB lie les caractères du texte en clair entre eux de manière à ce que le texte chiffré dépende de tout le texte en clair qui précède.

2.1.5 DES

Des est un algorithme à clé symétrique développé par IBM au début des années septante. Sa clé est de 56 bits de long, ce que la plupart des critiques actuels s'accordent à considérer comme trop peu.

Des est un codage de blocs CBC opérant sur 64 bit. Cette algorithme est très rapide grâce à sa clé très courte. Un PC basé sur un 80486 à 66 Mhz peut encoder jusqu'à 2,8 Mbit/s e logiciel, alors qu'un chip spécialisé peut dépasser (VLSI Technologies) les 512 Mbit/s.

On estime qu'il faudrait un million d'années à un Pentium 90 pour casser la clé avec une attaque en force brute.

A l'heure actuelle on emploie plus fréquemment un triple chiffrage DES (3DES), ce qui correspond à trois fois un chiffrage DES à 56bits.

2.2 Algorithmes à clés asymétriques

Les algorithmes à clé asymétrique ou clé publique, sont différents. Ils sont conçus de telle manière que la clé de chiffrement soit différente de la clé de déchiffrement. La clé de déchiffrement ne peut pas être calculée à partir de la clé de déchiffrement. Ce sont des algorithmes à clé public car la clé de chiffrement peut être rendue publique. N'importe qui peut utiliser la clé de chiffrement pour chiffrer un message mais seul celui qui possède la clé de déchiffrement peut déchiffrer le message chiffré.

La clé de chiffrement est appelée clé publique est la clé de déchiffrement est appelée clé privée. Dans les algorithmes à clé secrète, tout reposait sur le secret d'une clé commune qui devait être échangée dans la confidentialité la plus total, alors que la cryptographie à clé publique résout ce problème.

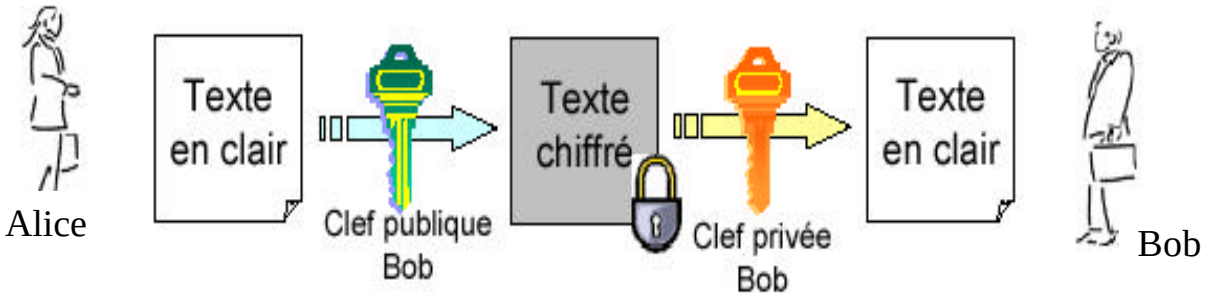


Figure 3 *clés asymétriques*

Sur ce schéma (Figure 3) on constate qu'Alice chiffre le texte à l'aide de la clé publique de Bob, Bob sera le seul à déchiffrer le texte car lui seul possède la clé privée associée.

La possibilité d'utiliser deux clés différentes pour traiter un message réside dans l'existence de fonction à sens unique.

2.2.1 Fonction à sens unique

Une fonction à sens unique est une fonction relativement aisée à calculer mais considérablement plus difficile à inverser. Dans ce contexte, « difficile » veut dire qu'il faudrait des millions d'années pour calculer la fonction inverse même si tous les ordinateurs du monde s'attelaient à la tache.

D'un point de vue mathématique, il n'y pas de preuve que des fonctions à sens unique existent ni même d'indice qu'elles peuvent être définies, mais cependant de nombreuses fonctions ont l'air d'être à sens unique. Par exemple dans un champ fini, il est facile de calculer le produit de nombre, mais la factorisation de ce produit en nombre simple est nettement moins évidente.

Un autre exemple utilisé pour de tels algorithmes est le problème des logarithmes discrets
Soit un grand nombre p, et un générateur g, et soit la relation suivante :

$$g^x = y \mod p$$

Calculer une exponentielle est facile, mais retrouver x en connaissant y revient à résoudre un logarithme discret, ce qui est extrêmement difficile.

A ce stade, de telles fonctions ne semblent pas avoir d'intérêt pour le chiffrement vu qu'il est impossible de les déchiffrer. Mais on définit une brèche dans la fonction à sens unique, un bon exemple d'une telle fonction est une branche d'arbre, depuis une feuille il est facile d'atteindre le tronc, il suffit de suivre la branche, mais depuis le tronc il n'est pas évident de retrouver la feuille. La brèche dans ce cas consisterait à connaître le chemin à suivre sur la branche.

Une fonction à sens unique à brèche secrète est donc facile à calculer dans un sens, quasiment impossible à calculer dans l'autre sens sauf pour celui qui connaît la brèche.

2.2.2 Fonction de hachage à sens unique

Une fonction de hachage à sens unique est légèrement différente d'une fonction à sens unique, une fonction de hachage à sens unique convertit une chaîne de caractères de longueur quelconque en une chaîne de caractères de taille fixe souvent de taille inférieure, cette chaîne est appelée empreinte (HASH). Le résultat d'une fonction de hachage est le même pour la même chaîne d'entrée, mais en principe il n'existe pas deux résultats semblables de fonction de hachage. Une exemple simple d'une telle fonction serait le byte résultant du XOR des bits d'une chaîne.

Mais étant donné que le résultat de la fonction a une longueur finie, il n'est pas possible de certifier qu'il n'existera pas deux valeurs d'entrées donnant le même résultat, dans un tel cas on parlera de collision, les algorithmes qui implémenteront des fonctions de hachage à sens unique viseront bien entendu à limiter de telle collision.

Une fonction de hachage est une fonction à sens unique car il est facile de calculer l'empreinte d'un chaîne mais retrouver la chaîne à partir de l'empreinte est quasi impossible. (Figure 4).

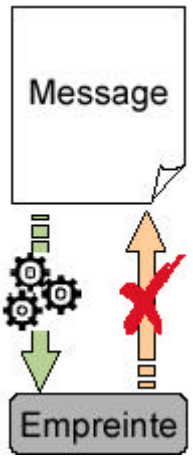


Figure 4 *fonction de hachage*

Les fonctions de hachage sont très utilisées pour vérifier l'intégrité d'un document. Le rédacteur du document passe celui-ci dans une fonction de hachage, puis transmet cette empreinte avec le document. A la réception, le destinateur pourra sans autre vérifier l'intégrité du document. Il suffira de repassé le texte dans la fonction de hachage, et de comparer l'empreinte obtenue avec l'empreinte fournie par le rédacteur.

Des fonctions de hachage sont également très utilisées pour le transfert de mot de passe sur le réseau.
L'utilisateur transmettra l'empreint de son mot de passe plutôt que le mot de passe en clair, le fichier de mot de passe du serveur réalisant le contrôle d'accès contient également que l'empreinte des mot de passe utilisateur.

Quiconque intercepterait la communication ne connaîtrait que l'empreint du mot de passe mais jamais le mot de passe car la fonction qui à généré l'empreint est à sens unique.

La fonction de hachage est publique car il n'y a pas de secret dans l'opération, elle est sûre, car elle est à sens unique, on ne peut pas retrouver l'entrée en connaissant la sortie. Il est ainsi possible d'associer une empreinte à un fichier, garantissant, comme une signature que le fichier est bien celui qu'il est sensé être.

2.2.3 Limitation de la cryptographie à clé publique.

Malgré l'aspect révolutionnaire de la cryptographie à clé publique, ces algorithmes ne peuvent pas se substituer aux algorithmes à clé secrète. Principalement pour une raison.

Les algorithmes à clé publique sont lents, généralement 1000 fois plus lent qu'un algorithme à clé secrète. De ce fait le chiffrement des messages ne se fait quasiment jamais sur la base d'un algorithme à clé publique, leurs usages étant confinés à la partie malgré tout très critique qu'est l'échange des clés.

Toutefois ils existe des algorithmes à clé publique qui peuvent être adaptés pour le chiffrement et la signature numérique..

2.2.4 RSA exemple d'algorithme à clé asymétrique.

Baptisé ainsi d'après le nom de ces créateurs. Ron Rivest, Adi Shamir et Leonard Adleman, il est le plus populaire des algorithmes à clé publique et aussi le plus simple à comprendre. Bien que les spécialistes n'aient jamais prouvé la sécurité ou la non-sécurité de RSA, cela inspire un certain niveau de confiance dans l'algorithme.

Le niveau de sécurité de RSA dépend de la difficulté à factoriser des grands nombres, les clés publiques et privées sont des fonctions d'une paire de grands nombres premiers. Retrouver le texte en clair à partir d'une des clés et du texte chiffré est supposé équivalent à la factorisation du produit de deux nombres premiers.

Pour générer les deux clés, il s'agit de choisir deux grands nombres entiers p et q. Puis de calculer le produit

$$n=pq$$

Ensuite on choisit un nombre e tel que e et $(p-1)(q-1)$ soient premiers entre eux, le nombre e est appelé clé de chiffrement aléatoire. Finalement on utilise l'algorithme d'Euclide étendu pour calculer la clé de déchiffrement d.

Cet algorithme permet de calculer d

$$d = e^{-1} \bmod ((p-1)(q-1))$$

La clé publique est formée par les nombres e et n, et la clé privée est le nombre d.
Pour chiffrer un message *M*, il suffit de résoudre l'équation

$$C=M^e \bmod n$$

Et pour déchiffrer

$$M=C^d \bmod n$$

Bien que la vitesse de l'algorithme puisse être améliorée en choisissant au mieux la valeur du nombre e, elle reste toutefois 1000 fois plus lent que les algorithmes à clé symétrique tel DES.

De plus les données à chiffrer doivent être au moins inférieures à la taille de la clé publique, une clé publique de 1024 bits ne peut chiffrer que des données de moins de 1023 bits.

Bien que cet algorithme ne semble pas rivaliser d'efficacité avec les algorithmes à clé symétrique, il n'en reste pas moins intéressant pour l'échange des clés et la signature numérique.
Ces deux notions seront vues plus en détail par la suite.

2.3 Échange de clés à l'aide de la cryptographie à clé publique

Il s'agit d'un système hybride, qui utilise la cryptographie à clé publique pour la négociation d'une clé de session commune qui sera utilisée pour le chiffrement des données, cette politique d'échange des clés est utilisée dans le protocole SSL.(voir 15.5.1)

Alice qui désire établir une communication sécurisée avec Bob génère une clé de session aléatoire et la chiffre avec la clé publique de Bob, en pratique les clés publiques sont disponibles dans une base de données comme LDAP.

Bob déchiffre le message à l'aide de sa clé privée, et connaît ainsi la clé de session commune. Alice chiffre ensuite le message avec la clé de session connue par bob qui pourra aisément le déchiffrer. (Figure 5).

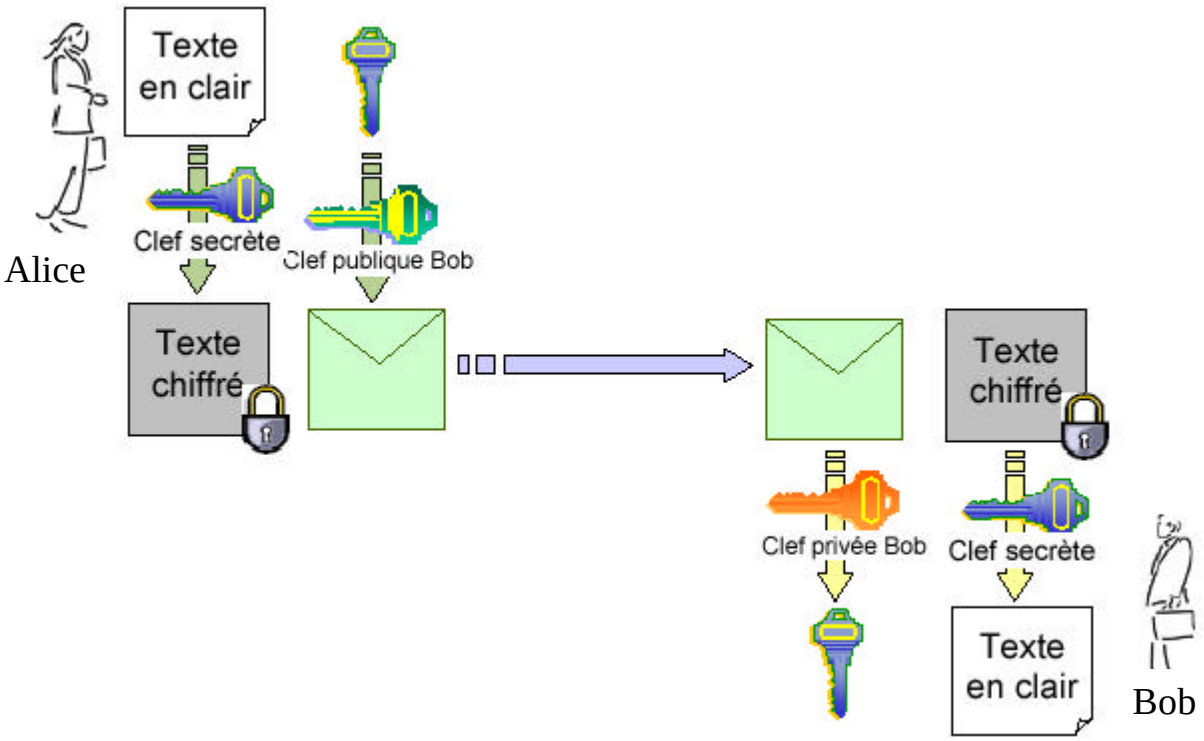


Figure 5 *Chiffrage hybride*

Mais cette méthode est sensible à l'attaque dite du **« men in the middle ».**

Son principe est le suivant :

Lorsque Alice interroge la base de données pour connaître la clé publique de Bob, Xavier, un adversaire puissant se positionne entre les deux tiers et intercepte la clé publique, il intervertit cette clé avec la sienne.

La clé de session générée par Alice sera chiffrée avec la clé publique de Xavier, il ne lui reste plus qu'à déchiffrer pour connaître la clé de session.

Ensuite il chiffrera cette clé avec la clé publique de Bob et lui transmettra le message. Par la suite, pour chaque message transmis, l'intercepteur procédera à son déchiffrement avec la clé correspondante puis le rechiffrera avec l'autre clé avant de l'envoyer à son destinataire : les deux tiers croiront communiquer de façon sûre alors que l'intercepteur pourra en fait lire tous les messages, voire même forger de faux messages.

Cette attaque est possible car les clés publiques de Bob et d' Alice ne sont pas authentifiées, c'est à dire qu'il n'y a pas de lien entre l'identité physique de ces personnes et leur clé publiques.

Une solution est de faire authentifier les valeurs publiques par une troisième personne de confiance, c'est ce qui sera décrit en détail dans le chapitre « 5 authentification à l'aide d'une tierce personne de confiance «

2.4 échange des clés par Diffie –hellmann

Inventé en 1976 par diffie et hellman les pères de la cryptographie à clé publique, cet algorithme est donc par la force des choses un algorithme basé sur une composition contenant une partie publique et une partie privée.

Le but de cet algorithme est que chaque entité puisse générer la moitié d'un secret et fournir à l'autre entité les paramètres permettant de calculer la seconde moitié du secret partagé, et ceci sans avoir aucune information préalable l'un sur l'autre.
Sa sécurité dépend de la difficulté à calculer des logarithmes discrets sur un corps fini.

Pour y parvenir l'algorithme est le suivant : (figure 6)

Bob et Alice se mettent d'accord sur deux grands nombres entiers n et g. Ces deux nombres doivent avoir les propriétés suivantes.

$(n-1)/2$ doit être premier, et de grande valeur.

Ces deux nombres doivent être tels que pour tous $b=1$ à $n-1$, il existe un a tel que

$$g^a = b \pmod n,$$

on dit que g est primitif à n.

Ensuite Bob va générer un nombre entier aléatoire b et envoyer à Alice le résultat du calcul suivant.

$$B = g^b \pmod n.$$

Alice à également générer un nombre aléatoire a et transmis à Bob.

$$A = g^a \pmod n.$$

Alice peut calculer le nombre $k = B^a \pmod n$ et Bob $k' = A^b \pmod n$ ce nombre est égal des deux cotés et définit le secret partagé, celui ci peut ensuite être utilisé pour dériver une ou plusieurs clés(clé secrète, clé de session, clé de chiffrement de clé).

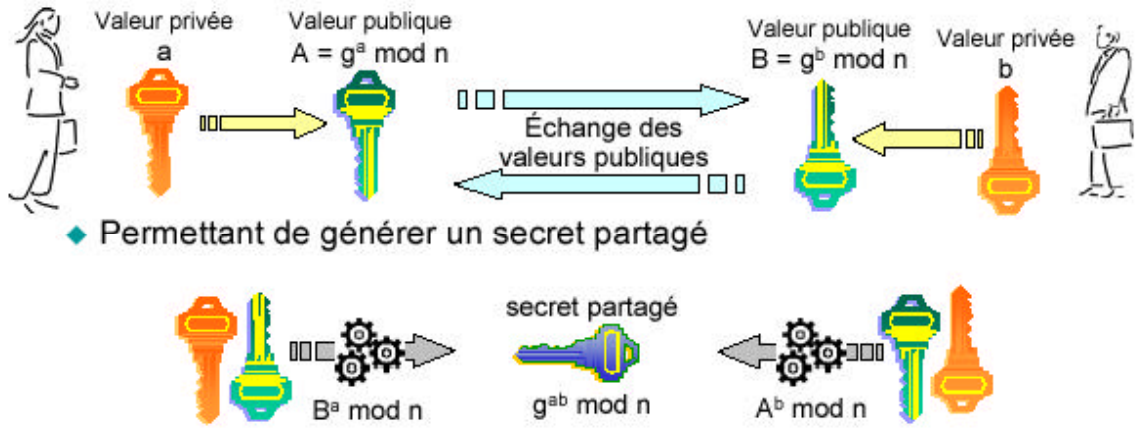


Figure 6 *Diffie-Hellmann*

La sécurité de cet algorithme est définie par le fait que quiconque aurait écouté la communication ne connaîtrait que `n,g,A,B.` Pour connaître `k`, le pirate devrait calculer des logarithmes discrets, ce qui est quasiment irréalisable si `n` est très grand.

Toutefois comme pour la remarque qui avait été faite concernant l'échange des clés par cryptographie à clé publique, cet algorithme est sensible à l'attaque de l'intercepteur.

Un adversaire qui se positionne entre les deux tiers et intercepte les échanges, peut de cette façon procéder à un échange de clés avec chaque tiers. A la fin du protocole, chaque tiers utilisera donc une clé différente, chacune de ces clés étant connue de l'intercepteur.

Une façon de contourner ce problème est d'authentifier les valeurs publiques utilisées pour la génération du secret. Deux approches peuvent être utilisées.

- En utilisant un service d'authentification des clés publiques, à l'aide de certificats numériques, PKI
- En signant les valeurs publiques avant de les échanger

Dans les deux cas, on perd néanmoins l'avantage de cet algorithme, qui a la possibilité de générer un secret partagé sans aucune information préalable sur l'interlocuteur.

3 Authentification.

3.1 But de l'authentification

Nous avons vu qu'il est possible de s'assurer de la confidentialité des données, mais cette confidentialité ne vérifie pas l'identité de votre interlocuteur, un intrus peut tout à fait se faire passer pour votre destinataire et ainsi usurper son identité à votre insu comme dans l'exemple du. **« men in the middle »** (2.3)

L'attaque du men in the middle est possible si aucune authentification n'a été entreprise. Avant de chiffrer des données il est nécessaire de s'assurer que la personne avec laquelle on communique et bien celle qu'elle prétend être.

Plusieurs méthode d'authentification sont possible. Il a été démontré qu'il existait des algorithmes symétriques et asymétriques pour chiffrer un message. De la même manière, il existe des algorithmes symétriques et asymétriques pour assurer l'authentification. La signature numérique est un procédé asymétrique alors que le scellement est symétrique.

3.2 Authentification asymétrique

Ce mode d'authentification se base sur l'utilisation de deux clés distinctes une clé privée et une clé public.

3.2.1 Signature numérique.

Une signature doit convaincre le destinataire que le document a bien été réalisé par celui ci, pour cela, elle doit être authentique est difficilement imitée. En principe une signature ne devrait pas être réutilisable, elle devrait faire partie intégrante du document ; de plus un document signé ne peut plus être modifié, le document signé est inaltérable.
Dans la réalité, on s'aperçoit que ces exigences sont en principe respectées, car il n'est pas évident de copier une signature manuscrite ; il n'en est pas de même pour des données informatiques car la présence d'une signature sur un document ne représente rien, vu la facilité avec laquelle un fichier peut être dupliqué et modifié.

3.2.2 Signature par la clé privée.

Il a été montré précédemment qu'il était possible de chiffrer un message de manière sûre avec la clé publique, et que seule la personne possédant la clé privée pouvait le déchiffrer.
Mais de cette manière, il est également possible de chiffrer un message avec sa clé privée, ainsi le message peu être authentifié avec sa clé publique, c'est-à-dire par tout le monde.

Chiffrer un document avec sa clé privée engendre une signature numérique sûre du document, car seul le propriétaire de la clé privée a été capable de le chiffrer.

Cette méthode est efficace car elle respecte les contraintes énoncées précédemment, l'authenticité est respectée. La signature est infalsifiable car c'est la clé privée qui la générée.
La signature n'est pas réutilisable car elle fait partie intégrante du document. Le document est immuable car la moindre falsification sur le document provoquerait un non déchiffrage du document.
L'algorithme à clé publique RSA permet d'effectuer de telles signatures

3.2.3 Signature par fonction de hachage et clé publique

Dans les applications pratiques, les algorithmes à clé publique sont souvent trop inefficaces pour signer de longs documents. Pour gagner du temps, les protocoles de signatures numériques sont souvent réalisées avec des fonctions de hachage à sens unique. Au lieu de signer le document, l'on signe l'empreinte du document (Figure 7). La vitesse de ce procédé est beaucoup plus élevée et comme les chances d'avoir deux documents différents ayant la même empreinte est très faible, signer l'empreinte est aussi fiable que signer le document tout entier.

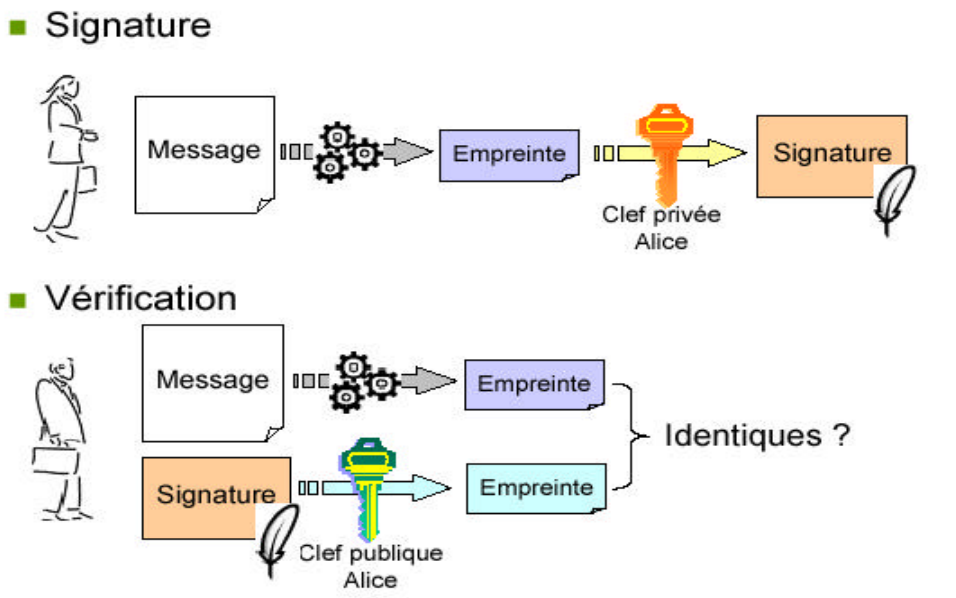


Figure 7 *Signature numérique*

En résumé, la personne dont on désire vérifier l'identité utilise un document dont nous avons une copie. Celui-ci calcule son empreinte à l'aide d'une fonction de hachage à sens unique, puis le chiffre avec sa clé privée.

Connaissant le document original, nous calculons son empreinte par la fonction de hachage, nous déchiffrons le document de l'émetteur avec sa clé publique, puis nous comparons celui-ci avec l'empreinte calculée, si l'empreinte est la même, c'est que l'identité de l'émetteur est correcte.

3.3 Authentification symétrique

3.3.1 Authentification par scellement.

Cette méthode consiste à adjoindre au message un sceau ou code d'authentification de message MAC (Message Authentification Code) qui est le résultat d'une fonction de hachage à sens unique à clé secrète. Tout se passe en théorie comme avec une fonction de hachage énoncée précédemment, sauf qu'il faut avoir la clé pour calculer l'empreinte. L'empreinte dépend à la fois des données et de la clé, elle n'est donc calculable que par les personnes connaissant la clé (Figure 8).

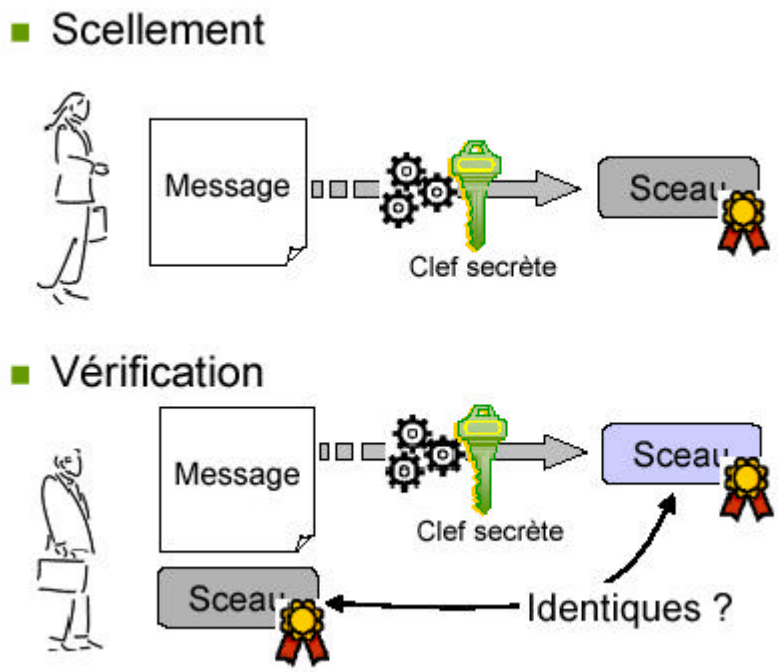


Figure 8 *Scellement*

Le scellement est une façon incontestable d'ajouter une authentification à un message, il est même plus rapide de sceller un document par une fonction de hachage à clé secrète que d'ajouter une signature numérique à celui-ci. De telle fonction sont appelées HMAC, il est possible de modifier les fonctions de hachage à sens unique conventionnelle en fonction de hachage à clé secrète, ainsi on trouve des fonctions HMAC-sha et HMAC-md5.

4 Échange de clés et authentification

Pour établir une communication sécurisée, la première étape consiste en une authentification à des fins de contrôles d'accès, on doit s'assurer que la personne avec laquelle on va échanger les clés est bien celle qu'elle prétend être et pas le « men in the middle ». Puis, on peut procéder à l'échange des clés proprement dit, la combinaison de l'authentification et de l'échange de clés est un échange de messages qui porte le nom de protocole d'authentification mutuelle avec échange de clé.

4.1 Définition des clés

Pour comprendre les différentes méthodes d'échange de clés, il est nécessaire de définir certaines clés ainsi que leur rôle dans les protocoles.

- **clé maîtresse** : Il s'agit de clé donc la fonctionnalité n'est pas de chiffrer, mais de créer par dérivation d'autres clés qui elles pourront servir au chiffrement ou a l'authentification.
- **clé de session ou de chiffrement** : Une telle clé sert par opposition à la clé maîtresse à chiffrer des données. Elles ont une durée de vie très courte, pouvant changer à chaque message. Ces clés sont souvent des clés symétriques car comme mentionné précédemment les algorithmes à clé symétrique sont nettement plus efficaces pour le chiffrement.
- **Clé de chiffrement de clé** : Ces clés ont une longue durée de vie et servent comme son nom l'indique, exclusivement à chiffrer d'autres clés. Ce sont très souvent des systèmes à clé publique qui sont utilisés pour le chiffrement de clé.

4.2 Propriété des protocoles d'échange de clés.

Pour qu'un protocole d'échange de clé soit sûr, il faut qu'il satisfasse aux deux conditions suivantes :

- Lorsque l'une des entités a accepté l'identité de l'autre entité cela signifie qu'aucun message n'a été altéré en route. Les messages sont donc semblables de part et d'autre.

- Il est matériellement impossible pour toute personne autre que les deux entités en présence de retrouver la clé qui a été échangée.

Ces deux conditions sont nécessaires, mais pas suffisantes pour assurer la fiabilité du protocole, d'autre propriétés sont souhaitables et sont notamment mises en évidence pour comparer les divers protocole qui seront décrit.

- La propriété dite de *Perfect Forward Secrecy* (PFS) est garantie si la découverte par un adversaire du ou des clés maîtresses ne compromet pas les clés de session générées précédemment : les clés de session créées ne pourront pas être retrouvées à partir des secrets à long terme. On considère généralement que cette propriété assure également que

la découverte d'une clé de session ne compromet ni la clé maîtresse ni les autres clés de session.

- La propriété dite de *Back Traffic Protection* (BTP) est fournie si la génération de chaque clé de session se fait de manière indépendante : les nouvelles clés ne dépendent pas des clés précédentes et la découverte d'une clé de session ne permet ni de retrouver les clés de session passées ni d'en déduire les clés à venir.
- On dit qu'il y a authentification directe (Direct Authentification) si, à la fin du protocole, les valeurs servant à générer le secret partagé sont authentifiées ou si chaque tiers a prouvé qu'il connaissait la clé de session. Par opposition, l'authentification est dite indirecte (Indirect Authentification) si elle n'est pas garantie à la fin du protocole.
- On parle de protection de l'identité (Identity Protection) lorsque le protocole garantit qu'un attaquant espionnant les échanges ne pourra pas connaître les identités des tiers communicants.

5 Authentification à l'aide d'une tierce personne de confiance

5.1 Signature de documents à l'aide d'un cryptosystéme à clé symétrique et d'un arbitre

Alice veut signer un document et l'envoyer à Bob, mais comme Bob n'est pas sûr de l'identité d'Alice, ils décident donc d'engager une troisième personne, Ivan qui aura le rôle d'arbitre.
Ivan est une personne de confiance, il n'essaiera jamais de profiter des informations qu'il possède à son profit.

Il partage une clé secrète `Ka` avec Alice, et une clé secrète `Kb` avec Bob, ces clés ont été créées bien avant qu'Alice ne veuille envoyer de document à bob.

La communication suit le déroulement suivant.

Alice chiffre sont message pour Bob avec la clé secrète `Ka` et envoie le résultat à Ivan, Ivan le déchiffre puis le complète en indiquant qu'il a reçu ce message d'Alice, puis le chiffre avec la clé qu'il partage avec Bob. Bob peut le déchiffrer et il certain qu'il vient bien d'Alice car il a confiance dans les dires d'Ivan.

Le problème de ce protocole est qu'il nécessite un travail intensif de la part de Ivan, en effet celui-ci doit systématiquement déchiffrer puis chiffrer les messages. De plus tout repose sur la confiance accordée dans ce participant intermédiaire.

5.2 Kerberos

Kerberos est un protocole d'authentification à tierce personne de confiance conçu pour les réseau TCP/IP. Un service Kerberos, résidant dans le réseau, agit comme un arbitre de confiance.
Kerberos est basé sur l'utilisation de la cryptographie à clé symétrique (DES en générale). Kerberos partage une clé secrète différente avec chaque entité du réseau, comme Kerberos connaît la clé secrète de tous le monde, il peut créer des messages pour convaincre une entité de l'identité d'une autre personne. Kerberos permet aussi de créer des clés de session qui sont données aux clients et aux serveurs, elles permettent de chiffrer les messages entre deux participants, ensuite cette clé de session est détruite.

5.2.1 Fonctionnement de Kerberos

L'agent de confiance Kerberos est représenté par Ivan, ce personnage en qui tout le monde a confiance. Ivan possède les clés secrètes de Alice et Bob. Alice désire engager une session avec Bob, elle envoie un message à Ivan avec son identité et celle de Bob.

Ivan engendre un message avec la datation, une longévité, une clé de session aléatoire, et l'identité d'Alice. Ce message est chiffré avec la clé secrète de Bob, puis ce même message est également chiffré avec la clé secrète de Alice. Ivan envoie les deux messages à Alice.
Alice déchiffre se message et extrait la clé de session, puis Alice engendre un message avec son identité et la datation, chiffre cela avec la clé de session fournie par Ivan et l'envoie a Bob. Elle envoie aussi à Bob le message qu'elle a reçu de Ivan chiffré avec la clé secrète de Bob. Bob déchiffre le message avec la clé qu'il partage avec Ivan et extrait la clé de session qu'il va partagé avec Alice, puis il déchiffre le message d'Alice. Bob engendre un message avec la datation plus un, le chiffre avec la clé de session et l'envoie à Alice, la communication est alors engagée.

L'utilité de dater les messages permet d'éviter qu'une demande soit rejouée ; ce protocole est efficace mais il présume que toutes les horloges sont synchronisées avec l'horloge d'Ivan ce qui n'est pas trivial.

5.2.2 Description générale Kerberos version 5

La version 5 de Kerberos diffère de la version 4 uniquement dans le contenu des messages, dans ce tutorial uniquement la version 5 sera décrite.

Un système Kerberos se compose de deux éléments, d'une part un serveur Kerberos et d'autre part un service de délivrance de ticket, les deux éléments communiquent par une liaison sûre.

Un client demande au serveur Kerberos un ticket pour accéder au service de délivrance de tickets (TGS Ticket-Granting-Service). Ce ticket est appelé TGT (Ticket-Granting-Ticket), Kerberos le chiffre avec la clé secrète du client. Le client demande ensuite au TGS un ticket pour un serveur particulier. Si le client a le droit d'accès à ce serveur, le TGS lui retourne le ticket demandé (Figure 9)

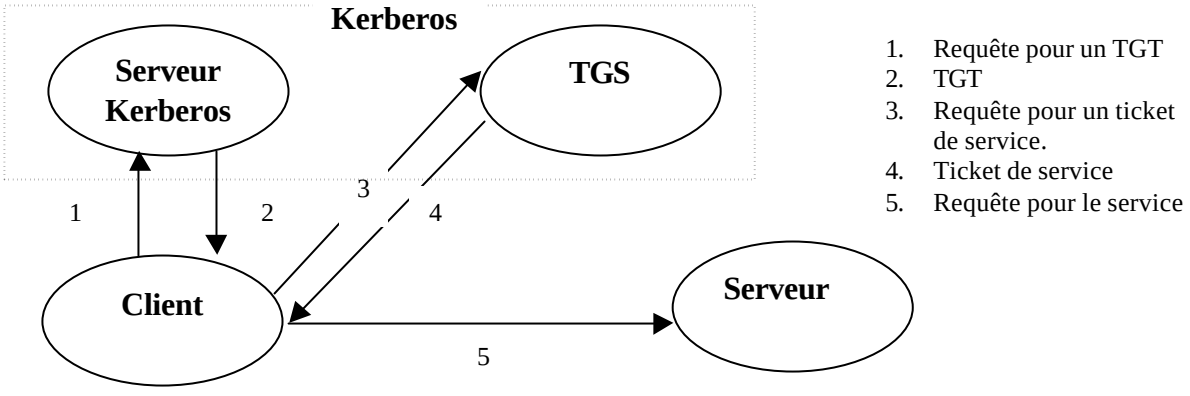


Figure 9 *Kerberos*

Un ticket de service est valable pour un seul serveur et un seul client. Il contient le nom du client, son adresse réseau, le nom du serveur, une datation et une clé de session. Il est chiffré avec la clé secrète du serveur. Le client ne peut évidemment pas déchiffrer ce ticket, mais il

l'utilise chaque fois qu'il désire accéder au serveur jusqu'à ce que sa date de validité soit échue.
Le serveur en recevant le ticket peut alors vérifier l'identité du client de façon sûre.

5.2.3 Description détaillée

Obtention du ticket TGT

Le client désirant obtenir un ticket TGT doit d'abord s'authentifier auprès de Kerberos, cette authentification se limite dans les cas les plus simples à la transmission du nom de l'utilisateur et d'un mot de passe.
L'agent d'authentification Kerberos cherche le client dans sa base de données, si le client est présent dans la base, il peut alors transmettre la clé de session qui sera utilisée entre le client et le TGS, cette clé de session est chiffrée avec la clé secrète du client, cette clé de session correspond au ticket TGT. Mais il est encore nécessaire que le client puisse s'authentifier auprès du TGS, pour cela Kerberos chiffre le TGT du client à l'aide de la clé secrète du TGS. Ces deux messages sont retournés chiffrés au client. Seul le client est en mesure de déchiffrer ce message.
Le client dispose alors de la clé de session qu'il va utiliser avec le TGS et également un moyen de s'authentifié auprès de celui ci.

Obtention de tickets pour un service

Jusqu'ici le client n'a de ticket que pour communiquer avec le TGS mais pas encore pour le service proprement dit.

Lorsqu'il a besoin d'un ticket pour un service particulier, le client chiffre sa requête avec la clé de session qu'il partage avec le TGS. Mais le TGS ne connaît pas encore la clé de session, c'est pour cette raison que le client doit transmettre également le TGT chiffré avec la clé secrète du TGS qui lui avait été fournie par le serveur Kerberos. Le TGS est en mesure de déchiffrer cette information, il extrait la clé de session et déchiffre la requête du client.

Si le client a les droits nécessaires pour le service demandé, le TGS lui fournit un ticket de service. Le TGS doit aussi créer une nouvelle clé de session qui sera utilisée entre le serveur et le client, celle-ci est évidemment chiffrée à l'aide de la clé de session partagée entre le client et le TGS.
Les deux messages sont transmis au client qui déchiffre le message et extrait la clé de session.

Demande de service

Maintenant le client est en mesure de s'authentifier auprès du serveur. Pour cela il crée un message très similaire à celui qu'il a envoyé au TGS (ce qui est logique puisque le TGS est un service).
Le client crée un message d'authentification composé de son nom, son adresse IP ainsi que d'une datation, le tout chiffré à l'aide de la clé de session entre le client et le serveur générée par le TGS. Le requête contient le ticket reçu de Kerberos (déjà chiffré avec la clé secrète du

serveur). Le serveur déchiffre le ticket et vérifie les informations comme la datation, l'adresse IP. Si tout concorde, le serveur sait que d'après Kerberos le client est bien celui qu'il prétend être.

Le client et le serveur chiffrent les futures messages avec la clé de session partagée.

5.2.4 Sécurité de Kerberos

Kerberos présente de nombreuses faiblesses au niveau de la sécurité. Steve Bellovin et Michael Merritt ont mis en évidence le problème pausé par la possibilité de rejouer des requêtes, bien que la procédure de datation ait pour but d'éviter cela, les messages peuvent être rejoué, pendant la durée de vie du ticket.

De plus Kerberos est sensible aux attaques dites de paris de mot de passe ; en effet, un intrus peut collectionner les tickets et ensuite essayé de les déchiffrer.

Mais l'attaque la plus sérieuse repose sur le fait que tous la confiance et mise dans le logiciel implémentant Kerberos. Rien n'empêche un utilisateur d'introduire un logiciel malicieux auprès de tous les utilisateurs, cette version se comporterait comme Kerberos mais permettrait de mémoriser tous les mots de passe.
Des améliorations de Kerberos sont à l'étude, comprenant une réalisation de cryptographie à clé asymétrique et une interface à carte à puce.

6 Public Key Infrastructure

6.1 Besoin d'un organisme de gestion des clés

L'utilisation massive de messages électroniques et l'expansion du commerce électronique dans le domaine professionnel comme privé est devenu une tendance de plus en plus populaire.

De ce fait, de plus en plus d'informations sensibles transitent par le réseau Internet, ces informations peuvent être sujet à diverse attaques malveillantes comme la célèbre attaque du « men in the middle » lorsque les intervenants échangent leurs clés publiques lors d'un cryptage asymétrique. (voire partie 2.3)

Dans une petite communauté, il pourrait être envisageable de générer sa paire de clés localement et d'échanger les clés publiques hors ligne, mais qu'en est-il pour une communication internationale où les échanges concernent des milliers d'utilisateurs.
Dans ce cas de figure, une authentification automatique des clés publiques est indispensable.

C'est dans ce contexte que la NIST (National Institute of Stantards and Technology) s'est vu imposer en 1994 la tâche d'étudier et de définir un standard dans la manière de gérer d'authentification les clés publiques pour le territoire des Etats-Unis en premier lieu, puis ce standard devait être étendu à un environnement international.

Ce projet avait pour but de permettre l'interopérabilité des différents systèmes électroniques opérant dans le commerce électronique. Le projet PKI (public Key Infrastructure) c'est construit autours des discutions et d'interviews effectués auprès de divers agence fédérales, comité de standard et d'organisation commerciale. L'étude à porté sur la manière de générer les clés, de les distribuer, d'obtenir les clés publiques au moyen de certificats, et la publication des certificats obsolètes communément appelé CRL (Certificate Revocation Liste).

L'étude visait à définir des recommandation techniques pour définir une architecture PKI au travers de divers composants qui partagent la responsabilité de la lourde tâche.

6.2 PKI définition

L'utilisation massive de la cryptographie à clé publique dans les échanges informatiques engendre un problème circonstanciel de taille, comme déjà mentionné.

Peut-on être sûr du propriétaire ou est-ce « man in the middle » ?

La PKI permet de résoudre se problème en permettant une authentification univoque des clés publiques.

A la façon d'un passeport ou d'une carte d'identité, la PKI va fournir une garantie d'identité numériques aux utilisateurs. Cette pièce d'identité numérique, appelée certificat numérique, contient la clé publique de l'utilisateur, mais également des informations personnelles sur

l'utilisateur du certificat. Comme tous document formel, le certificat numérique est signé par l'autorité de certification et c'est cette signature qui lui donnera toute crédibilité aux yeux des utilisateurs.

Mais contrairement à un passeport, le certificat numérique est largement publié, il n'a pas a être tenu secret, bien au contraire. Par exemple les browser Web permettent de stocker les certificats des sites Web et de tout autre utilisateur dans sa base de donnée interne.

Pour obtenir un certificat numérique, le client doit effectuer une requête auprès d'un organisme reconnu. Il transmet avec sa requête sa clé publique.

L'organisme construit un certificat incorporant la clé publique du client, il signe le certificat à l'aide de sa clé privée. (Figure 10)

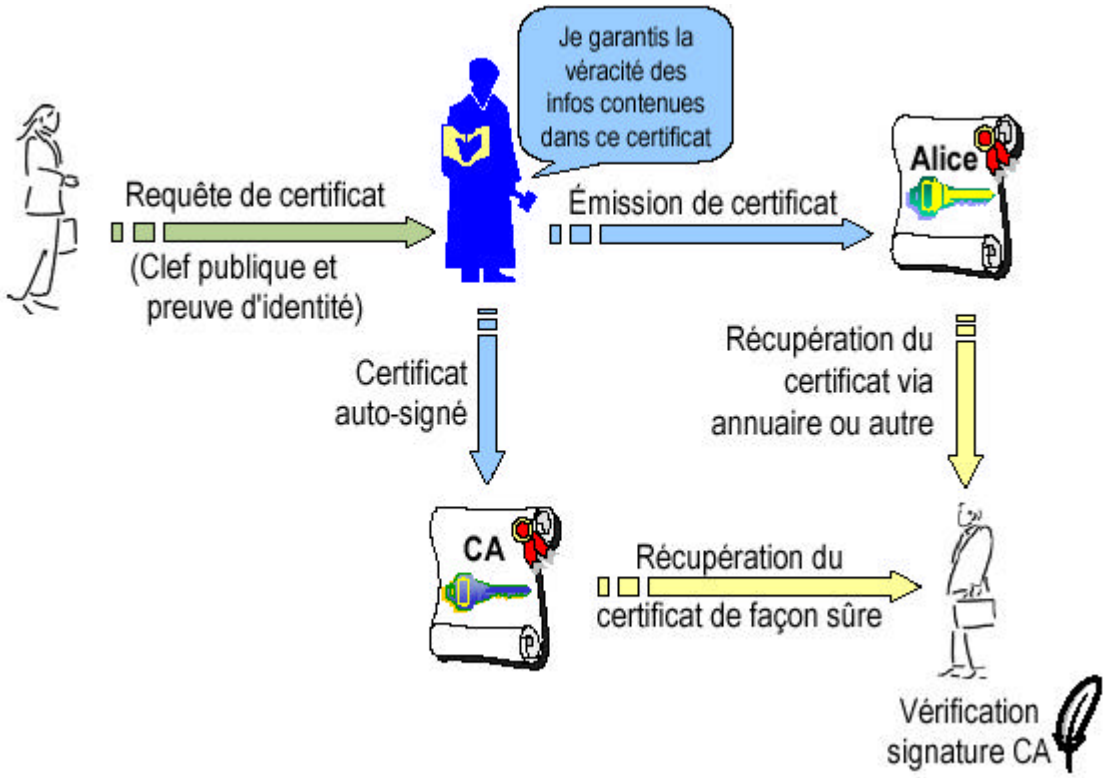

Je garantis la véracité des infos contenues dans ce certificat
Requête de certificat
(Clef publique et preuve d'identité)
Émission de certificat
Alice
Certificat auto-signé
Récupération du certificat via annuaire ou autre
CA
Récupération du certificat de façon sûre
Vérification signature CA

Figure 10 *PKI*

L'autorité de certification publiera le certificat signé comportant la clé publique et l'identité précise du propriétaire, quiconque consultera ce certificat aura l'assurance dans l'authenticité de la clé public contenue dans celui ci car il a confiance dans l'autorité de certification qui à délivré ce certificats. Par confiance il est entendu, que l'autorité est reconnu par l'utilisateur et que la clé publique de l'autorité soit préalablement connue.

6.3 Environnement sécurisé

Avant d'aller plus en avant dans la description des éléments constitutifs d'une PKI, il est primordial d'examiner les aspects de base de la sécurité, la sécurité au sens largue doit être étudiée avant de cibler le concept à la sécurité purement informatique.

La sécurité est un terme relatif, une sécurité n'est jamais absolue. Toute action quelle qu'elle soit représente un risque potentiel, malgré ça le risque est à la base du succès et de la productivité. Un environnement « absolument sans risque » est un environnement également sans potentiel.

C'est avec cette constatation qu'il s'agira d'opérer, pour définir et réaliser un système parfaitement utilisable en minimisant le risque.

Le design d'une solution sécurisée et à comparer à une défense militaire, le risque peut provenir d'une part des ennemis malveillants qui mettrons tout en œuvre pour pénétrer les défenses du système, mais le risque peut aussi provenir de l'intérieur soit par des collaborateurs sans scrupule soit par des accidents qui pourraient détruire l'infrastructure. A cet effet, une politique de sécurité physique doit être étudiée et définie.

6.3.1 Classification des ressources

Les utilisateurs de système sécurisé auront accès à différentes ressources, la première étape consiste à définir ces ressource et à les classer, en fonction de leur sensibilité.

Les utilisateurs devront être également classés, en plus de recevoir des badges et toutes sortes de laissez passer, il devront être classés en différentes catégories, (administrateurs, utilisateurs réguliers, utilisateurs occasionnels, etc), ceci permettra de différencier quant et à quelles ressources ces utilisateurs ont accès.

6.3.2 Séparer les zones publiques des zone privées

Une fois les utilisateurs et les ressources classées, il est indispensable de définir des séparations physiques pour affiner le contrôle. L'accès physique aux équipements doit être scrupuleusement contrôlé, par des moyens aussi simples qu'un local fermé à clé.

6.3.3 Protection contre les accidents

Un accident comme une inondation, un incendie, sont des incidents qui ne devraient pas paralyser complètement le système de sécurité. L'emplacement physique des ressources est à envisager avec le plus grand soin, des systèmes de reprise en cas de panne aussi bien qu'une répartition des équipements sur plusieurs sites peuvent se révéler judicieux dans ce type d'incidents.

Ces étapes sont le minimum pour garantir la sécurité physique des équipements et des données qui devront être mises en œuvre avant de définir la politique de sécurité intrinsèque à la PKI.

6.4 Gestion des clés

Les organismes d'infrastructure à clé publique ont besoin d'une discipline rigoureuse dans la gestion des clés, car il a été constaté qu'il est à l'heure actuelle beaucoup plus simple de s'introduire dans un système en se procurant les clés de manière illicite plutôt que d'essayer de casser un des algorithmes présentés dans le chapitre 2

Et un des instants les plus propices pour oser espérer se procurer les clés est sans conteste le moment où l'échange des clés aura lieu, il en résulte que cet échange doit être fait avec la plus grande prudence car il représente le point de voûte de tout le système.

La gestion des clés proprement dite se compose des opérations suivantes

- Génération
- Distribution
- Stockage
- Suppression.
- Archivage
- Recouvrement

6.4.1 Génération des clés

Il s'agit du moment où les clés sont initialement crées. Les clés sont générée de façon aléatoire, de manière à ce qu'elle soient non prédictibles. La prédictibilité dans le processus de création de clé peut être dévastateur en terme de sécurité, si le domaine de valeur dans lequel la clé va être défini est trop faible, tout le cryptosystéme est compromis. La génération de clés est donc une étape cruciale dans l'étude d'un système de sécurité.

Une clé symétrique n'est pas plus qu'un nombre aléatoire, générée soit par un générateur de nombres aléatoires ou pseudo aléatoire. Un générateur de nombres aléatoires est basé sur un algorithme hardware, alors qu'un nombre pseudo aléatoire est issu d'un algorithme logiciel qui prend comme paramètre un plus petit nombre comme base pour générer un nombre aléatoire (random seed).

La conclusion est qu'un nombre réellement aléatoire ne peut être généré par un algorithme logiciel, c'est à dire qu'un algorithme logiciel utilisant le même paramètre générera le même nombre aléatoire.

La génération des clés asymétriques est un processus plus complexe, il nécessite non seulement un nombre aléatoire, mais aussi un nombre premier et différents paramètres suivant

les algorithmes. Et nous savons que déterminer un nombre premier de grande taille est une tâche difficile.

Suivant l'application le processus de génération de clés doit être étudié avec la plus grande attention, et effectué sous contrôle.

6.4.2 Distribution des clés

La distribution est l'action de déplacer une clé de cryptage. Il existe deux étapes distinctes pour la distribution de clés, créer la clé initiale et créer les clés ultérieures. La clé initiale est établie et utilisée pour distribuer les autres clés.
La génération de la clé initiale ou maîtresse et une opération critique dans tout le processus de distribution des clés au sein de la PKI.

Si la clé initiale st sûre, elle peut être utilisée pour chiffrer d'autres clés qui seront distribuées par un canal moins sécurisé. Par exemple les clés de session sont très souvent chiffrées à l'aide d'une clé asymétrique, c'est notamment le cas pour SSL

6.4.3 Stockage des clés

L'étape qui suit la distribution des clés, est le stockage de la clé de façon sûre. La clé doit être protégée et doit garder à tout prix son intégrité et sa confidentialité. Le contrôle d'accès peut assurer l'intégrité et l'authenticité, mais seul un stockage sur support hardware peut assurer la confidentialité de la clé. Malheureusement à l'heure actuelle, les clés privées sont bien trop souvent uniquement protégée par un mot de passe utilisateur.

6.4.4 Suppression de clés

La suppression de clés intervient quand la clé à atteint sa fin de cycle, soit sa validité est à terme ou si une suspicion quant à la confidentialité de la clé pousse à la faire. La suppression signifie la destruction des toutes les copies de la clé symétrique pour un système symétrique et de la clé publique pour un système asymétrique. Une exception à cette règle intervient si le système dispose d'un processus d'archivage, dans ce cas les clés archivées ne sont jamais détruites.

6.4.5 Archivage des clés

L'archivage des clés permet de conserver une copie des clés même si elles ne sont plus utilisées, le but est de pouvoir valider des données qui ont été précédemment protégées par la clé. Toutefois des clés privées utilisées pour signer des documents ne devraient pas pouvoir être archivée car la sécurité des documents existants signés avec cette clé serait compromise.

Dans tous les cas, une clé archivée de peut pas être remise en service dans un environnement d'application.

6.4.6 Recouvrement des clés (Key Recovery)

Le recouvrement des clés est une procédure délicate qui permet de retrouver la clé privée d'un client. Elle peut être envisagée dans le cas où le client a perdu sa clé privée.
Un autre cas de figure peut apparaître si l'employé disparaît, soit physiquement par la mort de celui-ci, soit par la fin de son mandat de travail, toutes ses données encore chiffrées doivent pouvoir être recouvrées pour que son travail ne soit pas perdu. Dans ce cas, le principe de recouvrement de clés est souvent associé au recouvrement des données.

Un module de recouvrement de clés a pour but de stocker un double crypté de la clé privée de tous les clients dans un emplacement spécial. La procédure de recouvrement est une manœuvre lourde en conséquences, en effet le recouvrement ne doit pas avoir pour but un espionnage des données personnelles des clients, à cet effet la procédure de recouvrement de clé doit être impérativement menée par plusieurs personnes responsables, la procédure ne peut être effectuée qu'avec le consentement absolu de ces personnes.

Généralement le recouvrement de clés n'a lieu que pour des clés ayant servi à en crypter des données, les clés de signature ne peuvent en principe pas être recouvrée pour les mêmes raisons évoquées en (6.4.5)

6.5 Composant d'une PKI

La PKI est une association de plusieurs composants qui interviennent à différentes étapes mises en œuvre depuis la création du certificat jusqu'à la l'utilisation de celui-ci.

- Autorité d'enregistrement RA (Registration Authority)
- Autorité de certification CA (Certificate Authority)
- Application compatible PKI (PKI-ready)

6.5.1 Autorité d'enregistrement (RA)

Cette autorité à la tâche d'enrôler des nouveaux utilisateurs dans la PKI, elle reçoit des utilisateurs candidats les demandes de certificats CSR (Certificate Signing Request) ; elle à la responsabilité de vérifier la teneur de la demande.

Les méthodes de vérification utilisées dépendent de la nature du certificat demandé et de la politique de certification choisie. La vérification peut être limitée à l'identité du demandeur sur un formulaire HTML, mais on peut aussi vérifier si il possède bien la clé privée associée, s'il a bien l'autorisation de son organisation pour demander ce type de certificat, etc.

Les moyens mis en œuvre pour assurer cette vérification peuvent aller du simple échange de courrier électronique à une véritable enquête effectuée par les renseignement généraux.

Si la demande de certificats est acceptée, la demande est ensuite passée à l'autorité de certification CA qui n'a, elle, connaissance que des informations strictement indispensables à l'établissement du certificat. La requête est transmise suivant un format standardisé PCKS#10.

Il y a trois avantages à utiliser une autorité d'enregistrement indépendante de la CA au sein d'une PKI.

- Les centres d'enregistrement peuvent être distribué géographiquement. Par exemple les utilisateurs peuvent être enrôlés dans la PKI via des centres RA situés à Zurich, Paris ou Tokyo, alors que les certificats sont généré pour tous les utilisateurs depuis une CA établie au USA.

- Séparer les opérations effectuées par le RA et le CA permet de séparer le processus de requête du processus de génération proprement dit et de signature.

- La tâche d'enrôler un nouvel utilisateur peut être très fastidieuse, surtout si la politique d'enregistrement est stricte. En délèguent cette opération à une autorité autonome, on soulage l'organisme de certification de manière sensible.

6.5.2 Autorité de certification (CA)

Cette autorité est une autorité de confiance qui a pour but de créer les certificats des utilisateurs. La certification est l'opération qui consiste à lier l'identité d'un utilisateur à sa clé publique.

Le certificat généré contient entre autre le nom du demandeur Distinguished Name (DN), sa clé publique et une date d'expiration ainsi que la destination du certificat.

La date d'expiration stipule la durée de validité du certificat, alors que la destination précise dans quelle contexte sera utilisé le certificat, par exemple pour un serveur HTTPS ou pour une signature S/MIME.
Le CA signe finalement le certificat créé à l'aide de sa clé privée. Etant donné que tout le système PKI est basé sur une chaîne de confiance, la clé privée de la CA est un élément vital qui doit être protégé par tous les moyens, de ce fait la CA n'est pas nécessairement connectée à Internet. Dans des PKI de grande envergure, la CA est confinée dans un bunker protégé par des mesures exceptionnelles (blindage, contrôle de température, contrôle d'intrusion), celle ci est bien évidemment isolée complètement du réseau.

Suivant la politique de certification choisie, la CA peut prendre à sa charge une partie ou la totalité des opérations de la RA, c'est-à-dire vérifier l'identité de l'utilisateur et la teneur du certificat.

La CA garde une responsabilité sur la mise à jour des certificats qu'elle a générés, par exemple il est envisageable qu'un utilisateur change de secteur d'activité, rendant les informations contenues dans le certificat obsolètes, ou bien si l'utilisateur n'a plus confiance dans l'intégrité de sa clé privée La CA doit prendre en compte cette modification en révoquant son certificat quand bien même le certificat n'a pas atteint sa date d'expiration.

La CA doit donc transmettre la liste des certificats révoqués CRL de la même manière que les certificats générés. Les applications devront donc contrôler systématiquement cette CRL lorsqu'un certificat numérique leur est présenté.

6.5.3 Application compatible PKI (PKI-ready)

Un des plus grands avantages d'utiliser un PKI et plus particulièrement les certificats numérique pour l'authentification est que la norme est supportée par nombre d 'équipements et de logiciels.

- Web browsers
- E-mail
- VPN hardware et software

Les deux browser les plus communément utilisés qui sont Netscape Navigator et Microsoft Internet Explorer sont compatibles PKI. Ils permettent aux utilisateurs d'effectuer une génération de clés et un téléchargement de certificats numériques.

Les logiciels de traitement d'E-mail comme Microsoft Outlook et Netscape Messanger sont aussi compatibles PKI. Les utilisateurs peuvent signer leur courrier électronique par un simple clic de souris.

Grand nombre d'entreprises ont choisi une solution VPN pour interconnecter les différents réseaux. Le protocoles comme Ipsec utilise les certificats numérique pour authentifier les intervenants.

Les utilisateurs qui accéderont à ces applications présenteront leur certificat numérique, soit directement à l'application, soit à un gateway. Les applications vérifieront la teneur du certificat, d'une part à l'aide de la date de validité, puis en comparant la signature du certificat à l'aide des signatures de confiance déjà en leur possession, puis, suivant les cas, l'application vérifiera dans la CRL si le certificat n'a pas été révoqué.

Ces étapes correspondent à la procédure de contrôle effectué lors d'un payement par carte de crédit. validité, signature, révocation.

6.6 Répartition des CA

Les certificats générés pour la population de la terre ne peuvent pas être issus d'une même CA, il est donc nécessaire de repartir le travail à travers plusieurs CA. Dans le cadre d'une même PKI, il peut donc exister plusieurs CA effectuant le même travail. Quelle doit donc être la relation qui lie toutes les CA ? (Figure 11)

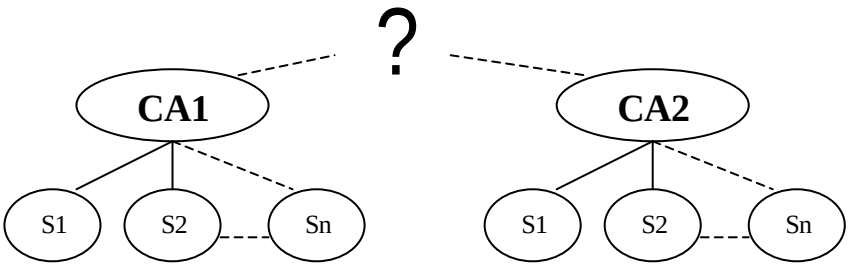


Figure 11 *Multi CA*

Chaque CA va générer des certificats S1, S2, Sn, étant donné que les CA n'ont pas de relation directe, pour valider un certificat issu de chaque CA, les utilisateurs devraient disposer des clés publiques des différents CA. Ce modèle structurel de répartition des autorités n'est donc pas envisageable.

6.6.1 Modèle hiérarchique

Le modèle hiérarchique présenté sur la figure suivante (Figure 12) permet de résoudre ce problème.

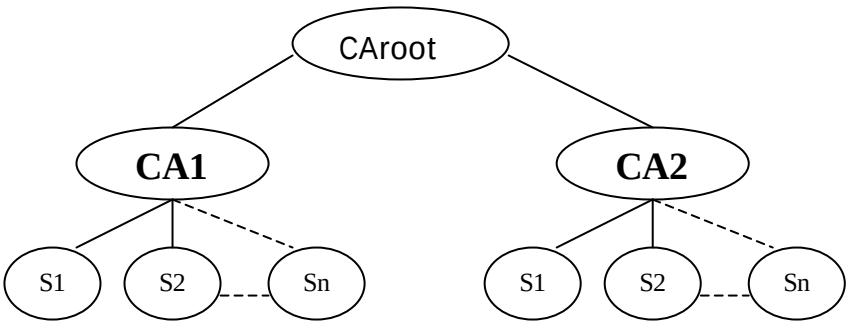


Figure 12 *Ca root*

Les autorités CA1 et CA2 ont soumis leurs clés publiques à un Ca Root qui leur à généré un certificat. L'autorité Ca Root peut être défini comme le plus haut niveau d'autorité ; c'est le seul composant qui ait un certificat auto signé. Un certificat auto-signé est le seul certificat qui permette d'assurer l'intégrité mais pas l'authenticité.

CA1 et CA2 deviennent des CA subordonnées.

Ce modèle hiérarchique définit une relation entre le CA root et les CA subordonnés, une chaîne de confiance est ainsi établie. Les utilisateurs ont confiance dans le CA root, mais par la définition de la chaîne de confiance, ils ont également confiance dans les CA subordonnées.

Etant donné que tout le système repose sur la confiance accordée au CA root, il est primordial que sa clé privée soit maintenue dans un endroit « absolument sur ». Le CA root représente un point de faiblesse potentiel de toute la PKI, si la clé privée du CA root venait à être compromise, tous les certificats généré par les CA subordonné deviendraient suspect avec toutes les implications dramatiques que cela produirait, tous les certificats devraient alors être retirés.

Les CA subordonnés peuvent également avoir sous leur responsabilité d'autres CA et ainsi de suite, augmentant sensiblement la complexité du modèle . Le modèle hiérarchique impliquant un CA root n'est pas la seule architecture possible.

6.6.2 Modèle Peer to peer

Dans ce modèle (Figure 13), les CA travaillent au même niveau hiérarchique, un ou plusieurs CA peuvent générer des certificats de manière croisée dans la relation peer to peer, les certificats ainsi générés portent le nom de certificat co-signé ou co-certifié

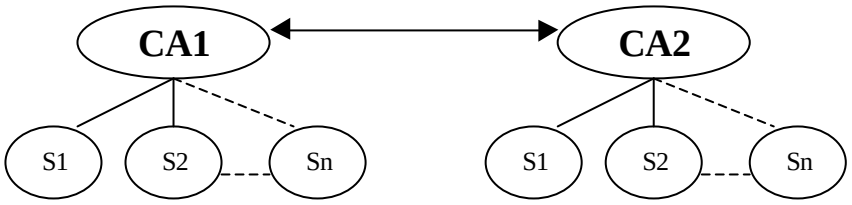


Figure 13 *Co-certification*

Les deux CA s'échangent mutuellement leurs clés publiques ; elles sont alors en mesure de générer un certificat pour leur homologue. Dans ce schéma, CA1 voit CA 2 comme son root et réciproquement il n'y a donc pas de point de faible unique. Toutefois étant donné que CA1 est responsable de l'authenticité de CA 2 il se porte garant de tous les certificats délivrés par CA 2, ce qui n'est pas une mince affaire suivant les politiques de certification.

6.6.3 Modèle en pont

Le modèle hiérarchique a été jugé trop restrictif, ce qui a impliqué qu'aucune agence gouvernementale ne voulait porté la responsabilité d'être le CA root pour toutes les autres organisations. Le modèle de certification croisée, quant à lui, est difficile à mettre en œuvre lorsque le nombre de CA augmente, en effet pour **N** autorités de certification il fallait générer $\mathbf{N^2 - N/2}$ certificats pour certifier toutes les autorités.

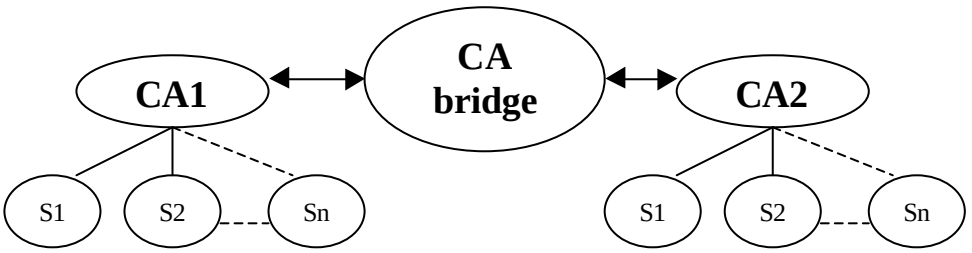


Figure 14 *CA Bridge*

Dans ce modèle (Figure 14), CA1 et CA2 n'échange leurs clés publiques qu'avec le CA bridge, les échanges de certificats croisés suivent une complexité en **N** au lieu de $\mathbf{N^2}$ pour le modèle sans pont.
La politique de certification des CA doit être similaire afin d'assurer la compatibilité du modèle ; cette remarque concerne bien évidemment le modèle de certification croisée précédent.

6.7 Politique de certification

La politique de certification décrit l'ensemble de la procédure qui conduit à certifier une clé publique. Cette politique prend en considération les moyens mis en œuvre pour vérifier les informations constituant le certificat et la destination de celui-ci (certificat personnel pour la signature S/MIME, certificat de serveur HTTPS, etc).Une autorité de certification peut appliquer plusieurs politiques de certification selon les populations et les usages concernés.
Quelques exemples de politiques de certification :

• *Thawte personnal freemail* : certificat gratuit obtenu quasiment sans vérification, le seul élément de preuve de l'identité du demandeur est acquis par échange de mot de passe par courrier. On a donc la preuve que le demandeur a accès à la boite aux lettres dont il prétend être titulaire.

• *Thawte personnal basic* : il faut 100 points pour obtenir ce certificat. Les points sont obtenus auprès d'un «notaire Thawte»qui peut vous attribuer 50 points si vous vous présentez physiquement avec une copie de votre pièce d'identité. Avec 150 points vous devenez notaire. Cet exemple de politique de certification est assez intéressant ; si trois complices malhonnêtes prouvent leur identité, ils peuvent devenir notaires et dès lors enregistrer de fausses demandes de certificats (Thawte propose un troisième niveau de certification *«Thawte personnal premium»).*

• *Swisskey :* ce service suisse vous demande de présenter une pièce d'identité en cours de validité auprès d'un représentant de son autorité d'enregistrement c'est à dire auprès de certaines banques (milieu dans lequel on a l'habitude de vérifier l'identité des personnes).

Ces exemples montrent bien que la solidité des algorithmes de chiffrement ou la longueur des clefs utilisées est relativement secondaire devant les aspects organisationnels de la PKI. L'IETF a défini dans le RFC-2560 un formalisme de description d'une politique de certification.

6.8 Le certificat X509

Les utilisateurs de certificats étant de plus en plus nombreux, le format de ce certificat doit de ce fait être commun à tous les utilisateurs. Sans cela, il serai impossible d'intégrer ces certificats dans des applications logicielles développées par des différents fournisseurs, pour cette raison, les certificats numériques sont soumis à un standard.

Le certificat X509 fait l'objet d'une normalisation par l'ISO. Il a été réalisé par l'IETF (Internet Engineering Task Force)et est identifié par un «Distinguished Name »(DN). C'est concrètement un document électronique attestant qu'une clé publique est bien liée à une organisation, une personne physique, etc. Historiquement les certificats étaient utilisés pour protéger l'accès à des annuaires de type X500. De ce fait, la structure d'un certificat X509 se reflète à travers ses composants, le lien entre le nomenclature X509 et X500 est flagrant.

Ce document électronique contient une clé publique, un certain nombre de champs à la norme X509 et une signature. C'est la liaison des attributs des champs et la clé publique par une signature qui constitue un certificat. Un certificat peut être faux; c'est sa signature par une autorité de certification (CA)qui lui donne une authenticité.

• Globalement, la composition d'un certificat x509 est la suivante (Figure 15):

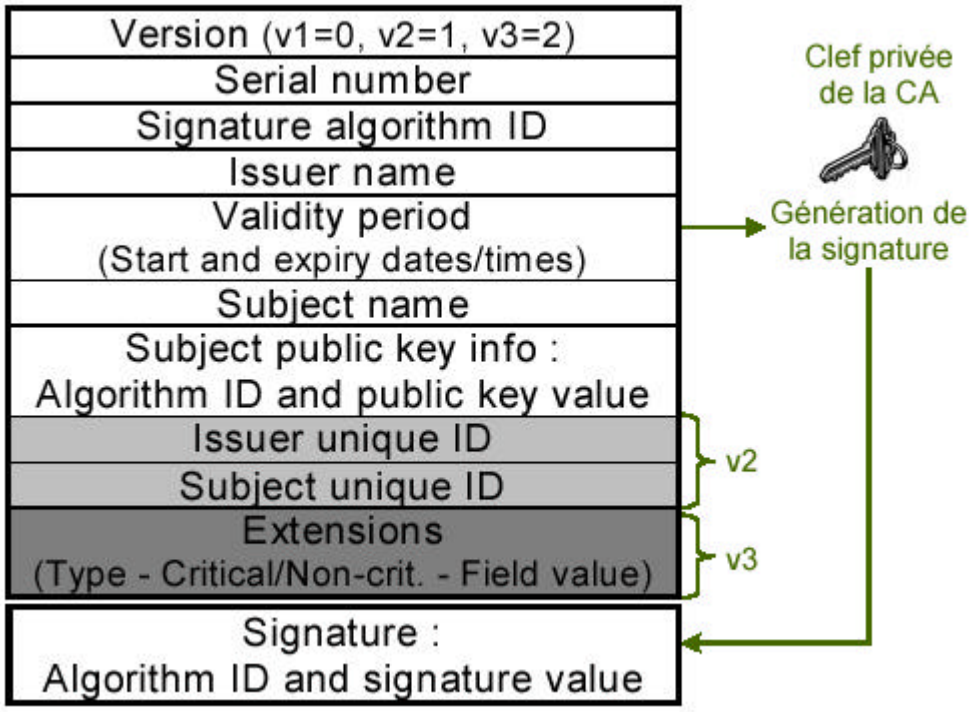


Figure 15 *certificat X509*

- ***Version :*** Ce champ identifie à quelle version de X.509 correspond ce certificat
- ***Serial number*** : Numéro de série du certificat (propre à chaque autorité de certification).
- Signature Algorithm ID: algorithme utilisé pour signer le certificat.
- ***Issuer Name:*** Distinguished Name (DN) de l'autorité de certification qui a émis ce certificat.
- ***Validity period***: C'est une paire de date pendant laquelle le certificats est valide.
- ***Subject Name*** : Distinguished Name (DN) du détenteur de la clé publique.
- ***Subject public key info*** : Le nom de l'algorithme à clé publique (ex RSA), ainsi que tous les paramètres concernent cette clé, et la clé proprement dite.
- ***Issuer Unique ID/Subject Unique Id*** : Extensions optionnelles introduites avec la version 2 de X.509
- ***Extensions*** : Extensions génériques optionnelles, introduites avec la version 3 de X.509
- ***Signature*** : Signatures numériques de la CA sur l'ensemble des champs précédents

Les extensions apportées par la version 3 du standard X.509 permettent de coupler un type et une valeur. Un paramètre supplémentaire «témoin » permet de déterminer si l'extension doit être prise en compte. Les extensions permettent de définir des profils de certificat.

- Banques
- Organisation public

- Associations
- Etc.

6.9 Service de révocation CRL

Un certificat numérique, comme une carte de crédit doit pouvoir être révoquée si un changement d'identité du propriétaire a lieu, ou si la clé privée de l'utilisateur est perdue ou divulguée. Les certificats ne peuvent pas être détruits ou retirés car leur présence peuvent apparaître à des milliers d'endroits chez les participants PKI.

Dans ce cas, le service de révocation mis en œuvre par le CA doit enregistrer la demande de révocation et vérifier son authenticité. L'auteur de la demande est-il bien la personne titulaire de la clé publique ? Une fois la vérification effectuée, la liste des certificats révoqués est publiée.

Il appartient aux clients utilisateurs de vérifier les listes de révocation pour les certificats qu'ils utilisent. La révocation est un élément du service de publication.

L'accès aux listes de révocation peut être spécifié dans le certificat sous forme d'une URL. Les clients peuvent alors téléchargés la liste de révocation CRL. Mais étant donné que cette liste est générée périodiquement par la CA, son utilisation n'est pas optimale car les utilisateur doivent mettre à jour constamment cette liste. De plus, si une CA met à jour sa liste CRL et révoque un certificat juste après, les utilisateurs ne verront cette modification qu'après la prochaine mise à jour de la liste, cette à dire le lendemain dans certain cas, cette politique n'est pas sans risque en terme de sécurité.

Pour contrer cet inconvénient les utilisateurs doivent disposer de la liste de révocation en temps réel, en vérifiant ces informations directement dans la base de donnée de la CA, cette vérification est possible par l'intermédiaire d'un élément OCSP (Online Certificate Status Protocol) qui se chargera d'interroger la Ca sur la validité d'un certificat.
http://www.certco.com/pdf/OCSP_Salz.pdf

De ce fait, la liste de révocation de la PKI est le seul élément devant disposer d'un service d'annuaire obligatoirement connecté à Internet.

6.10 Service de publication

Le service de publication permet l'accès des utilisateurs aux certificats des correspondants afin d'en extraire la clé publique.
L'utilisation du service de publication n'est pas requise pour toutes les applications de chiffrement asymétrique. En particulier, l'accès à un serveur HTTPS dans le but de chiffrer les échanges ou d'authentifier le serveur ne requiert pas un accès au service de publication car

le serveur HTTPS communique lui-même son certificat du serveur HTTPS. De même, il est possible d'échanger des messages S/MIME sans utiliser le service de publication (l'envoi d'un message signé permet de faire parvenir automatiquement au correspondant son certificat). Toutefois, l'utilisation du service de publication est un élément déterminant dès que le nombre d'utilisateurs augmente. L'identité de la personne certifiée est définie dans un Distinguished Name, elle constitue donc une clé d'accès dans l'annuaire LDAP. Par ailleurs LDAP est la seule API normalisée et donc utilisable dans le contexte hétérogène d'Intenet.

7 Annuaire

7.1 Annuaire et PKI

Souvent le service d'annuaire est mentionné dans le même cadre que la PKI. Les systèmes implémentant une PKI disposent également d'un système d'annuaire permettant la publication des certificats, mais ces deux entités ne sont absolument pas dépendantes l'une de l'autre. Lors de l'implémentation d'une solution PKI, le choix du modèle d'annuaire associé n'est pas une tâche aussi simple qu'elle n'y parait. Il s'agit de comparer la performance, la flexibilité du modèle, les outils de management à disposition et l'interopérabilité avec d'autres services d'annuaire.

7.2 Annuaire définition

Les annuaires constituent l'endroit où sont déposés les informations. L'information est organisée de façon logique afin de faciliter au maximum leur accès. Dans le domaine populaire, l'annuaire est l'équivalent des pages jaunes. Il contient les noms des compagnies commerciales et la façon de les contacter. Dans le domaine informatique, les annuaires contiennent les informations concernant les systèmes, les services réseau et les utilisateurs. De ce fait, un service d'annuaire peut être très simple, mais également devenir d'une grande complexité suivant la nature des informations contenues.

L'accès à l'annuaire est une opération qui doit être sécurisée, en effet il est nécessaire d'authentifié le client et de déterminer quelles sont ses privilèges avant d'effectuer sa requête sur l'annuaire.

L'annuaire est comparable à une base de données dans son fonctionnement. Toutefois, l'annuaire et la base de donnée différent sur plusieurs points :

- La base de données est sujette à une modification fréquente de ces informations, et ceci de manière complexe. Le résultat d'une requête de mise à jour dans une base de données peut avoir un effet sur des milliers d'enregistrements en même temps. Pour conserver les contraintes d'intégrité dans un tel cas, il est nécessaire de mettre en œuvre un système de gestion de transaction relativement puissant. Dans le cas de l'annuaire, au contraire, les données sont consultées plus souvent qu'elles ne sont modifiées, ce qui à permis de simplifier nettement le modèle, l'optimisant pour la lecture de l'information.

- La nature des informations contenues dans une base de données pousse à conserver en tout temps une contrainte d'intégrité sévère sur celle-ci ; alors que les informations contenues dans un annuaire sont moins sensibles et permettent une mise à jour plus souple de l'information.
  L'exemple suivant vous convaincra. Le fait qu'un employé venant de quitter son entreprise ait encore accès à son e-mail pendant plusieurs heures, porterait moins à conséquence qu'une perte de mise à jour de quelque heures dans une gestion de stock.

- Dans une base de données, des copies sont générée pour effectué un back up et conservé l'intégrité de la base en cas de panne, alors que dans un service d'annuaire, les données peuvent être dupliquées pour permettre un accès simultané par plusieurs utilisateurs dans un environnement distribué ; la mise à jour de celle-ci peut se faire de façon plus ou moins simultanée car l'intégrité de celle ci n'est pas garantie, comme mentionné précédemment.

7.3 Rôle de l'annuaire dans une PKI

Les composants critiques définis dans une PKI nécessitent un stockage organisé et une facilement accessibilité. Le service d'annuaire peut participer à cette tâche en assurant une organisation adéquate des données de la PKI est permettre son accès de façon simple. Bien que l'annuaire ne soit pas la seule manière de gérer cette tâche, elle est souvent privilégiée car elle permet d'utiliser un annuaire déjà en activité.

Le service d'annuaire est utile dans le cas d'une PKI pour différente raisons :

- Les certificats générés par une PKI peuvent être stockés dans l'annuaire et récupérés facilement par les utilisateurs et les applications.

- L'annuaire peut stocker également la liste de révocation CRL, permettant ainsi au utilisateur de vérifier la validité d'un certificats de façon simple.

- Les organisation PKI qui permettent de gérer le recouvrement de clé, peuvent utiliser l'annuaire pour stocker les clés privées, cryptées bien évidemment.

Le principe de recouvrement de clé peut être utilisé pour permettre aux utilisateurs mobiles de retrouver leur clé privé lorsqu'ils changent d'ordinateur ou de terminal. Traditionnellement, la clé privée est stockée de façon sûr et locale sur le disque dur de l'utilisateur. Les utilisateurs qui n'ont pas de poste fixe sont alors défavorisés. Les PKI mettant en œuvre un service de recouvrement de clé privée disponible dans un annuaire permettent un déploiement vraiment mobile des applications. Cette méthode de recouvrement n'est pas identique à la lourde procédure explicitée en 6.4.6

Pour être compatible avec la PKI, l'annuaire doit répondre à deux critères :

1. L'annuaire doit supporter le standard X.509v3 et permettre de stocker des CRL.
2. L'annuaire doit supporter le protocole LDAP(Lightweight Directory Access Protocol) le standard pour l'accès aux données par annuaire.

7.4 Protocole d'accès au répertoire

Pour avoir une vision plus pertinente du concept de d'annuaire, il est nécessaire de connaître les bases sur les protocole d'accès au annuaire que sont X .500 et LDAP.

7.4.1 X.500

X 500 est certainement le plus ancien modèle d'accès au annuaire. Il définit d'une part le protocole d'accès proprement dit DAP (Directory Access Protocol) et d'autre part le modèle d'information qui stipule comment celle-ci est stockée et gérée au sein de l'annuaire.

Le projet original X.500 était pour le moins ambitieux : il avait pour objectif de définir une infrastructure unique et publique d'annuaire qui décrivait complètement les ressources de la famille OSI, et contenir tous les outils nécessaires pour rechercher et naviguer parmi ces objets.

Le standard X.500 ne devint jamais le standard pour les applications Internet pour plusieurs raisons.

1. L'inconvénient majeur de X.500 est sa complexité excessive, souvent jugée trop « lourde »

2. Le standard X.500 est basé sur le modèle OSI, qui est difficilement supporté par les application Internet.

7.4.2 LDAP

Pour spécifier un standard d'accès aux annuaires pour le réseau Internet, le protocole LDAP fut créé en 1997 à l'université du Michigan.

La version LDAPv3 est définie dans le RFC2251.
LDAP tourne sur le standard TCP/IP ; il est très nettement moins lourd en ressource que son prédécesseur X.500. Cette raison à permis d'amener très rapidement ce protocole au niveau d'un standard d'Internet.

Actuellement, les compagnies développant des services d'annuaire pour Internet sont contraint à rendre leur produit pleinement compatible avec ce nouveau standard qu'est LDAP.

Quelques réserves sont toutefois à émettre sur ce protocole.
LDAP apporte la flexibilité est la simplicité, mais c'est au détriment de l'implémentation parfois fastidieuse pour de larges applications.

LDAP spécifie uniquement l'interface extérieure des clients, et ne définit pas la façon interne de gérer l'annuaire. Ce qui implique qu'il n'y ait pas de format standard quant à l'implémentation des fonctionnalisés de l'annuaire. Les différences d'implémentation peuvent conduire à des incompatibilités entre des systèmes de différents fournisseurs.

L'interopérabilité étant primordiale dans le cadre d'une PKI, il est essentiel d'étudier la compatibilité entre les différents serveurs LDAP utilisés, dans le cas ou ces systèmes proviendraient de différents fournisseurs.

7.4.3 Architecture LDAP

Bien que le contrôle d'intégrité sur les donnés contenues dans un annuaire soit moins strict que pour une base de données proprement dite, il n'en est pas moins qu'un mécanisme de contrôle doit exister.

Ces mécanismes doivent définir quel type de données est autorisé, comment celles-ci peuvent placées dans l'annuaire, et comment on peut y accéder.

L'unité de base d'information stockée dans l'annuaire est une entrée (entries). L'entrée peut être une personne, une entreprise, un certificat X.509, etc. Chaque entrée est identifiée de manière univoque par son DN (Distinguished Name).
Une entrée est composée d'attributs contenant les informations sur l'objet en question. Le type de l'attribut est défini par un nom et une référence sur un objet OID (Object Identifier). Les attributs et leurs OID sont standardisés de façon univoque dans le schéma de l'annuaire.

Les entrées sont conservées dans l'annuaire dans une structure en arbre DIT (Directory Information Tree).(figure 16)

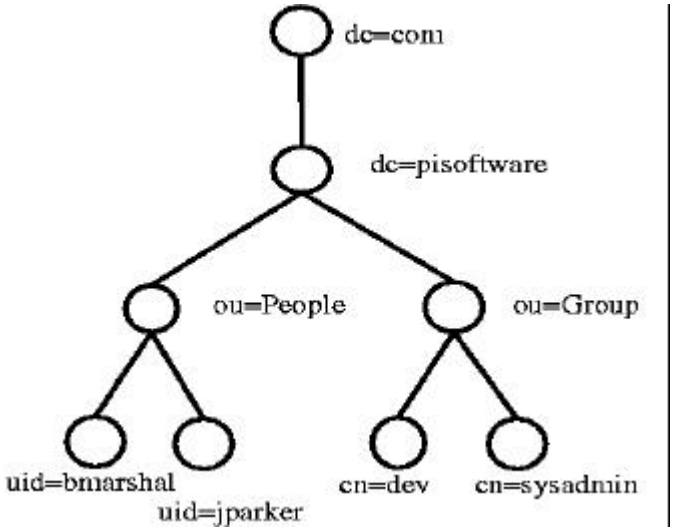


Figure 16 *Hiérarchie LDAP*

L'organisation du modèle de données doit être mis en place le plus scrupuleusement possible car il est la pièce maîtresse du service d'annuaire.

Le modèle doit être ouvert et extensible de manière à pouvoir être modifié au besoin lors de la croissance de l'entreprise. Le schéma doit aussi être flexible pour permettre une interopérabilité avec des modèles d'autre organisation.

7.4.4 Sécurité d'accès à l'annuaire

A l'heure actuelle il n'est pas déraisonnable de penser que la puissance est donnée à celui qui détient l'information. Les PKI mette à disposition des informations dans des annuaires afin d'apporter aux clients une sécurité dans leurs transactions. Les annuaires contenant de l'information doivent alors être protégés comme tout équipement PKI contre des attaques potentielles.

Les requêtes effectuées et les réponses fournies en retour par l'annuaire doivent absolument être protégées au niveau transport. A cette effet LDAPv3 supporte SSL.

Les clients qui accèdent à l'annuaire peuvent être protégés par un nom d'utilisateur et un mot de passe, ou utiliser une authentification plus forte par l'intermédiaire de certificats ; mais étant donné qu'un des rôles de l'annuaire est justement de distribuer ces certificats, de tels types d'authentification ne peuvent pas toujours être mis en œuvre.

Pour limiter au maximum les accès à des données sensibles, l'annuaire doit être partitioné. Ainsi, dans une partition, les clients n'auront de visibilité que sur une fraction de l'annuaire ou des données. Le partitionnement permet de définir une zone publique qui contient des informations non confidentielles que tout un chacun peut visualiser, tout en garantissant la confidentialité sur les données plus sensibles.

Il s'agit aussi de limiter les privilèges de chacun sur l'annuaire. Ainsi, un utilisateur client de la PKI ne pourra avoir accès aux données qu'en lecture seule, alors qu'un administrateur aura le droit accordé de modifier le contenu de l'annuaire.
Le contrôle d'accès est assuré par l'implémentation d'une liste de contrôle ACL (Access Control List) qu'il revient à chaque constructeur d'implémenter car l'ACL n'est pas définie dans le standard LDAP.

8 Protection des clés privées

8.1 accès aux clés privées

Toute la philosophie d'une architecture à clé publique repose sur la confidentialité de la clé privée des utilisateurs et surtout celle de l'autorité de certification. Si votre clé privée est volée, non seulement vos communications pourront être décryptées, mais de faux documents pourront être signés à votre insu, ce qui conduit à une situation désastreuse dans un environ ment ou les transactions électroniques sont fréquemment utilisées. La clé privée est l'élément le plus sensible de tout le mécanisme d'infrastructure à clé asymétrique.

Dans la plupart des cas la clé privée est stockée sur le disque dur . Or il a été démontré (par van Someren and Shamir en 1998) que la clé privée avait une caractéristique tout à fait significative. Elle comporte de longue plages de bit à valeur aléatoire comparativement aux autres données. Autrement dit, rechercher des plages de bit aléatoire sur le disque dur pourrait amener à trouver la clé privée. Pour pouvoir effectuer une telle recherche il était alors bien sûr nécessaire d'avoir un accès direct à votre disque dur. Donc, de telles attaques ne pouvaient être pratiquées que pendant votre absence « lunch time attack ». http://www.simovits.com/archive/keyhide2.pdf

Pour éviter de laisser sa clé publique dans le système à tout moment, la solution d'une clé privée stockée sur un support hardware mobile à été énoncée.

Les modules matériels qui permettent de contenir la clé privée sont appelés HSM(Hardware Storage Module), ils respectent un standard de sécurité définit par le NIST, leurs formes peuvent varier, périphériques, carte PCMCIA, smart cards .

Dans tous les cas, la clé privée est générée à l'intérieur du HSM est n'est jamais extraite telle quelle de ce support. Les données qui nécessitent un décryptage ou une signature numérique sont passées au HSM par une interface standard.

Tout le processus cryptographie est effectué à l'intérieur du module. Ce processus permet de ne jamais laisser le logiciel utiliser la clé privée de façon directe.

8.2 Smart Cards

La smart card est un équipement matériel qui présente des caractéristiques intéressantes. Sa taille compacte lui donne une apparence similaire à une carte de crédit ; son aspect familier lui confère une grande crédibilité aux yeux de la population pour un déploiement à largue échelle.

Contrairement à une carte de crédit, elle n'a pas de bande magnétique, qui est le point faible notoire des cartes bancaires d'ancienne génération. La smart card est l'équivalent HSM adapté pour la PKI, elle comporte une puce à microprocesseur et un certaine quantité de mémoire lui permettant de stocker les clés et les certificats
Mais son microprocesseur interne constitue également un moteur cryptographique hardware, utilisable aussi bien pour les algorithmes symétriques qu'asymétriques.

Les standard d'interfaces pour les smart cards sont.
PKCS#11 ; PKCS#15 ; SO7816;Microsoft CryptoAPI et le standard PC/SC.

Les applications compatibles smart card sont en pleine expansion, Web browser authentification, wireless communications, contrôles d'accès, signature numérique.

Mais la récente loi approuvée concernent la crédibilité apportée par la signature numérique dans le commerce électronique global, est certainement la fonctionnalité qui propulsera la smart card dans le standard de masse, comme l'avait été la carte de crédit de son temps.

9 Politique de sécurité

9.1 Références légales

Les organisations implémentent des PKI pour résoudre les problèmes de sécurité réseau, toutefois la sécurité réseau n'est qu'une partie de la politique de sécurité totale de l'entreprise, comme mentionné précédemment, le risque d'infiltration physique n'est jamais nul.

Les applications PKI opèrent dans un cadre de travail ou les responsabilités sont répartie suivant des critère légaux et sociaux qui sont définie dans un cadre plus largue qui est la politique de sécurité.

Avant de mettre en œuvre une politique de sécurité réseau, les organisations doivent définir les privilèges des utilisateurs, le rôle des administrateurs, le système de maintenance et comment l'organisation va prévenir et réagir aux éventuelles brèches dans la sécurité. Cette politique est la ligne conductrice qui sera suivi lors de la mise en œuvre de la PKI.

Une fois cette problématique définie, la politique de sécurité réseau peut être définie, elle inclut couramment.

- **Rapport pratique de certification** Certification Practice Statement **(CPS)**
- **Politique du certificat** Certificate Policy**(CP).**
- **Considérations légales**

9.1.1 Rapport pratique de certification (CPS)

Il s'agit d'un document légal, créer et publier par l'autorité de certification, il spécifie les critère de certification et la politique de révocation des certificats. Ce document sera jugé par les utilisateurs pour définire le niveau de confiance qu'ils peuvent placé dans l'autorité de certification.

9.1.2 Politique de certificat

Une politique de certificat a pour but d'expliciter et de limiter l'utilisation du certificat numérique. En d'autre terme, elle définit le niveau de confiance qu'un utilisateur peut placer dans le certificat d'autrui. Ces indications peuvent être incluse à l'intérieur du certificat ou indirectement référencée.

9.1.3 Considération légal

Les organisations doivent déterminer qui est responsable en cas de perte ou de fraude à l'intérieur même de la PKI. Par exemple est ce qu'une CA porte l'entière responsabilité en cas de perte d'un certificat ? Ou est ce que la responsabilité est partagée entre divers élément de la PKI ? Une fois ces questions résolue, par la définition des droits et obligation de chaque entité. La politique de sécurité peut être mise en place.

10 PKI et authentification bio métrique

10.1 bio métrie définition

L'être humain comportent des caractéristiques physiologiques et physiques qui permettent de l'authentifier de manière univoque. La biométrie est la discipline qui utilise ces différences biologique pour déterminer, vérifier et identifié un individu. Les contrôles bio métriques principaux se basent sur les empreintes digital, reconnaissance vocal, scan faciale, scan de l'iris, géométrie de la main. Le but est de retirer de ces caractéristiques biologiques le minimum d'information afin de générer un échantillon unique, cet échantillon sera comparé avec la mesure effectuée lors de chaque contrôle d'identité

Il a été mis en évidence dans le chapitre 8 (protection des clé privée), le besoin évident de protection des clé privées. La faille d'un système de sécurité réside trop souvent dans la partie déléguée au client c'est à dire.

- Choix d'un mot de passe
- Protection de la clé

Si la protection des clé privée peut être assuré par un support de stockage hardware, le choix du mot de passe est toujours laissé à l'utilisateur.
La plupart des utilisateurs, si il n'ont pas été suffisamment été sensibilisé, choisiront un mot de passe excessivement simple, cette simplicité peut être exploitée par des intrus pour casser le mot de passe par force brute en utilisant des dictionnaires.

La bio-métrie permet de limité ce risque en utilisant une caractéristique physique comme mot de passe utilisateurs, ne laissant donc plus aucune responsabilité à l'utilisateur.

Une caractéristique biologique ne peut pas être perdue ou dérobée, ce qui est le principal avantage d'une authentification bio métrique. De plus le degré de confiance d'une telle authentification peut atteindre 100% suivant les technologies.

Les techniques d'authentification bio métriques se sont largement déployées dans le domaine des PC et de la sécurité d'accès des entreprises, il est donc tout a fait pensable d'adapter ces techniques pour les besoin de la PKI, précisément pour remplacer le mot de passe utilisateur.

Les méthodes d'authentifications bio métriques permettent d'atteindre un niveau de flexibilité inégalée par les technique traditionnelles .
La vérification fournit un résultat qui indique le degré de similarité entre l'échantillon stocké et la valeur mesurée. Le seuil de similarité peut donc être réglé afin d'obtenir un niveau de confiance aussi élevé que nécessaire.

10.2 Choix d'une technique bio métrique dans le cadre d'une PKI

La technique de vérification par lecture d'empreinte digital présente des qualités appropriées dans le cas d'une PKI.

- La détection d'intrus, qui essaieraient de s'introduire est facilement détectable.
- Une personne autorisée est rarement rejetée par ce type d'authentification
- Le coût de mise en œuvre est faible.

Toutefois la vérification par empreinte digital est difficile en cas de coupure ou suivant l'état des doigts, la situation démographique peut donc réduire sensiblement les performances du système.

La lecture d'iris permet de réduire nettement ces inconvénients, mis à part certaine maladie, la composition de l'iris reste stable. Les inconvénients de la méthode est son coût de mise en œuvre important et son déploiment difficile. A l'heure actuelle il n'est pas envisageable de mettre en œuvre de telle mécanisme de contrôle à largue échelle, d'une part par ce que les utilisateurs sont septique sur l'emploi de tels équipements et d'autre par ce que son utilisation ne peut être faite en présence d'un agent de contrôle.

La reconnaissance vocale comme la signature manuscrite sont des moyen de contrôle dont le taux de rejet est trop important, ce qui a souvent comme conséquence une certaine frustration de l'utilisateur rejeté par son propre système.

Une solution mis en œuvre par certaine compagnie dans le cadre du droit d'accès, et une combinaison de plusieurs facteurs bio métriques simultanés, comme contrôle facial plus reconnaissance vocal, ou empreinte digital et contrôle facial. Ces technologies combinée permet de réduire le risque de rejet tout en garantissant une authentification efficace à moindre coût.

A l'heure actuelle les techniques bio métriques sont soit mal adaptées soit trop coûteuses pour répondre aux besoins immédiat de la PKI. Mais ces technologies étant comme la PKI, en pleine évolution, il est fort probable que dans un avenir proche la PKI puisse bénéficier de la puissance apportée par la biométrie dans son processus d'authentification éliminant ainsi le risque du mot de passe utilisateur trop simple ou de la perte de sa carte à puce.

Ces techniques en collaboration devrait permettre d'apporter une sécurité parfaitement adaptées au besoin de toute entreprise.

Conclusion

Ce document n'a pas la prétention d'être exhaustif sur la question des PKI, mais il permet néanmoins d'apporter une vision globale et superficielle concernent la problématique de l'authentification et de l'échange des clés dans divers environnement sécurisé

De nombreux ouvrage de référence sur la sécurité informatique ont été publiés. Ces documents touchant aussi bien le coté technique que le coté juridique du sujet sont référencés en annexe.

Mais il faut garder à l'esprit que la sécurité absolue n'existe pas. Pour chaque données sensible, il est nécessaire de définir le prix que l'on est prêt à payer pour sa sécurité.

Plus les moyen mis en œuvre sont conséquent, plus les individus capable d'entreprendre une attaque sont rare. Toutefois les failles ne sont pas toujours la ou les prévoit.

Certaine suspicion pesse sur un ou des organismes particulièrement puissant ayant les moyens d'introduire des portes de faiblesse mathématiques, au sein même des algorithmes cryptographiques.
.
Mais c'est avec cette réalité qu'il s'agit d'opérer !

Questions de révisions

1. **Pourquoi la cryptographie à clés asymétriques ne peut elle pas remplacer complètement la cryptographie symétrique ?**

2. **Les algorithmes asymétriques utilisent fréquemment des clés de longueur de 1024 voire 2048 bits alors qu'un algorithme symétrique de 56 bits est jugé sûr. D'où vient cette différence ?**

3. **Quels sont les mécanismes mis en œuvre par les algorithmes symétrique pour résoudre la contrainte de diffusion nécessaire à l'aboutissement d'un chiffrage sûr ?**

4. **A l'aide d'un schéma, représentez les différentes étapes nécessaire pour signer un document.**

5. **Pourquoi signe t'on le résultat de la fonction de hachage et non pas le document lui même ?**

6. **A l'aide d'un schéma, représentez les différentes étapes nécessaire pour contrôler l'intégrité d'un document. (contrôle de la signature)**

7. **D'un point de vue conceptuel, quelle est la différence majeur entre une signature manuscrite et numérique ?**

8. **A l'aide d'un schéma représentez l'attaque du « men in the middle » dans un échange de clés publiques.**

9. **Sur quel type de cryptographie se base Kerberos (sym,asym)?**

10. **Sur quel type de cryptographie se base PKI ?**

11. **Pourquoi à t'on besoin de s'authentifier?**

12. **Pourquoi l'adresse IP d'un utilisateur ne suffit elle pas à prouver son identité ?**

13. **Dans une structure PKI qui génère les clés privées et publiques ?**

14. **Dans un système PKI supportant le key recovery, quelles sont les clés générées par l'utilisateur et quelles sont les clés générées par la PKI ? Pourquoi cette distinction ?**

15. **Lors que vous recevez un certificat numérique d'un site HTTPS, quel équipement client contrôlera la signature du certificat ?**

16. **A l'aide d'un schéma, représentez les différentes étapes qui peuvent intervenir pour contrôler un certificat numérique.**

17. Lorsque deux PKI décident d'adopter une hiérarchie croisée peer to peer, qu'advint t-il de la politique de certification propre à chaque PKI ?

18. Citez les avantages à utiliser un support hardware pour stocker les clés privées plutôt que le disque dur.

19. Pourquoi les certificats numériques délivrés doivent absolument être publié dans un annuaire ?

Bibliographie

Image et schéma :

[1] http://hsc.fr clip-art sécurité
[2] http://www.sopers.co.nz/master_key/master_key_systems.htm image de titre

Cryptographie :

[3] Bruce Schneider ; cryptographie appliquée ; International Thomson publishing 1997
[4] William Stallings ; cryptography and network security, prentice-hall ; 1999
[5] Adi Shamir and Nicko van Someren ; Playing hide and seek with stored keys
http://www.simovits.com/archive/keyhide2.pdf

PKI et x509 :

[6] Tom Austin ; PKI a wiley tech Brief ; wiley 2001
[7] A pratice guide to public key infrastructure ; xcert 2001
http://www.xcert.com/~marcnarc/PKI/thesir
[8] x509 présentation
http://www.hsc.fr/ressources/presentations/pki/img8.html
[9] C.Broillet ; Les Pki présentation ; eivd 2000
[10] George Mason ; Binding identities and attributes using digitally signed certificates
[11] rssi ; La pki test du cru ; 2001
http://pki.cru.fr
[12] What is a PKI?
http://www.rsasecurity.com/rsalabs/faq/4-1-3.html
[13] MITRE ; Public key infrastructure study final report ; 1994
[14] Conventional public key infrastructure : An Artefact Ill-fitted to the Needs of the Information Society
[15] The risk of key recovery, key escrow & trusted third party encryption
http://www.cdt.org/crypto/risks98
[16] Key escrow : its Impacts and alternatives
http://notwired.lycos.com/clipper/diffie.html
[17] Infrastructures de Gestion de clefs
http://www.urec.cnrs.fr/securite/articles/CA.CNRS_Test.pdf
[18] Deploying OCSP Responders
http://www.certco.com/pdf/OCSP_Salz.pdf
[19] X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP
http://www.ietf.org/rfc/rfc2560.txt

Ldap :

[20] Marcel Rizcallah ; Construire un annuaire d'entreprise avec LDAP ; eyrolles 2000
[21] OpenLDAP 2.0 Administrator's Guide
http://www.openldap.org/doc/asmin/

[22] Ldap howto
http://www.grennam.com/ldap-howto.html
[23] Missana Carole ; LDAP et OpenLDAP ; eivd 2001
[24] M.Jaton ; Service de téléinformatique ; eivd 2000

Biometrics and hardware support :

[25] Mini key PKI token
http://www.arx.com/assets/minikey _proddesc.pdf
[26] Authenticating with one of the safest devices the biometric 2001
http://www.securecomputing.com/pdf/sony *puppy* wp.pdf
[27] L.Reinest & S.Luther ; Biometrics, Tokens, & Public Key Certificates ; ISSO
http://www.biometrics.org/REPORTS/cert.pdf

11 Étude d'une PKI commerciale

11.1 Généralités

Le produit commercial étudié est le produit KEON développé par RSA Security.
Ce produit clé en main permet d'apporter une solution efficace en terme de sécurité pour toutes les applications PKI-ready, comme les VPN, ERP, mail-securitsé, etc.

Il fournit une famille de produits qui permettent de composer différentes gammes de solutions pour mettre en œuvre une architecture à clé publique PKI, basée sur la cryptographie asymétrique. Toute les potentialités du logiciel n'ont pas pu être testées, seul le module de base a été installé et testé, les commentaires concernant les modules additionnels proviennent de la documentation RSA.

11.2 But de l'étude

Le but de cette étude n'est pas de faire l'inventaire des fonctionnalités du produit KEON dans un contexte commercial. Il s'agit d'étudier les mécanismes et les outils fourni par ce produit dans l'optique de définir un cahier des charges des fonctionnalités indispensables qui devront être retrouvées dans un univers open source. Cette étude permettra de comparer de façon pertinente une implémentation d'une PKI commerciale et une solution PKI libre et gratuite.

11.3 Installation

La mise en œuvre du produit KEON est extrêmement simple. Tout est basé sur des interfaces HTML. L'installation du produit consiste à suivre progressivement les consignes fournies sur ces pages HTML. La configuration des différents serveurs formant la PKI sont effectuées suivant une procédure parfaitement automatique et transparente pour l'utilisateur.

Durant cette étape, différents certificats sont créés, comme un certificat root CA et un certificat CA subordonné. Un certificat administrateur donnant la totalité des droits sur la PKI est également généré.

11.4 Architecture PKI de KEON

Le logiciel Keon décompose les différents secteurs d'activité PKI en plusieurs serveurs, Secure administration, enrollment, directory et logging serveurs (Figure 17). Ces serveurs peuvent être géographiquement dispersés.
Il fournit également un moteur cryptographique puissant pour la signature numérique des certificats utilisateurs.

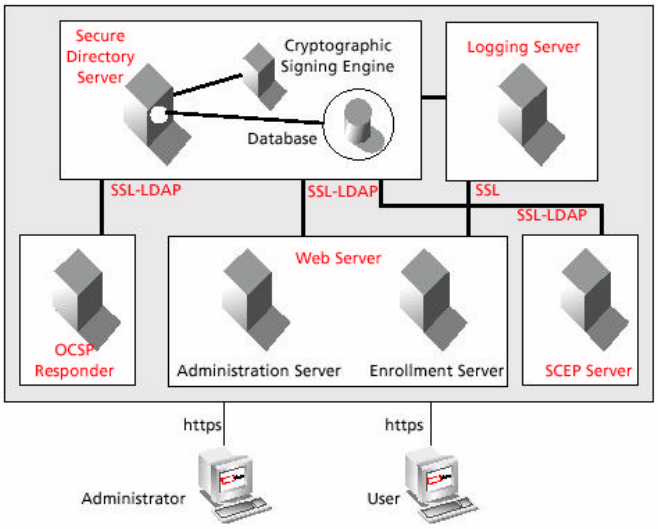


Figure 17 *Architecture KEON*

11.4.1 Serveur d'administration (Administration Server)

Le serveur d'administration PKI n'est accessible que par des administrateurs disposant d'un certificat numérique spécifique à leur fonction dans la PKI. Depuis ce serveur, l'administrateur dispose d'un contrôle total ou partiel sur les différents mécanismes permettant la gestion de la PKI.

Le serveur d'administration permet :

- De définir et de contrôler les droits d'accès des utilisateurs et des différents privilèges administrateurs.
- De gérer toute la hiérarchie PKI, c'est à dire construire une hiérarchie CA, créer ou recréer des certificats CA.
- De manipuler, contrôler et archiver toutes les requêtes de certificat transmis par les clients. Les actions sur les requêtes sont les suivantes : approuver, révoquer et suspendre les requêtes, mais également définir différentes extensions à adjoindre aux certificats utilisateurs. Cette manipulation proprement dite peut être effectuée par plusieurs administrateurs suivant leur fonction dans la PKI. La gestion de la PKI est répartie suivant

les rôles des administrateurs, un rôle spécifie des privilèges et des droits d'accès sur le serveur.

11.4.2 Serveur d'enrôlement (Enrollment Server)

Ce serveur d'enrôlement est la partie visible de la Pki par tous les utilisateurs finaux. C'est à ce serveur que les clients vont s'adresser pour solliciter un certificat. Les clients rempliront un formulaire qui sera transmis et contrôlé par les administrateurs du serveur d'administration.

L'aspect enrôlement du produit KEON fournit tous les mécanismes pour une sécurité maximum. La paire de clés RSA est générée par le client qui peut stoker directement la clé privée sur support hardware comme des smart card ou smart token. Il sera avisé par mail lors que son certificat sera approuvé.

11.4.3 Serveur des répertoires sécurisés (Secure Directory Server)

Ce serveur fournit une base de données où sont conservés tous les certificats, requête de certificat et liste de contrôle d'accès (ACL) de la PKI.
Ce serveur peut être accessible par les applications PKI-ready en lecture seul, par le protocole LDAP. Les serveurs PKI qui effectuent des modifications sur ce composant, utilisent LDAP protégé par SSL, nécessitant l'authentification par certificat numérique.

Ce serveur est physiquement intégré sur le même poste que le moteur cryptographique. L'accès à ce poste doit donc être parfaitement protégé.
Pour cette raison KEON permet l'utilisation de support hardware comme HSMs pour garantir une sécurité absolue dans le gestion de la clé privée utilisée pour la signature de certificat.

11.4.4 Logging server

Ce serveur est accessible par les trois autres serveurs pour enregistrer l'activité des administrateurs dans leurs différentes zones de contrôle.
Ce serveur va permettre de visualiser l'activité des différentes entités PKI, en vue d'une amélioration de la structure PKI.

11.5 Architecture CA

Le produit Keon permet de définir plusieurs autorités de certification subordonnées à la CA root, suivant le modèle **hiérarchique** (Figure 18)
Cette possibilité est nécessaire lorsque l'activité de la Pki devient trop conséquent pour une seul CA.

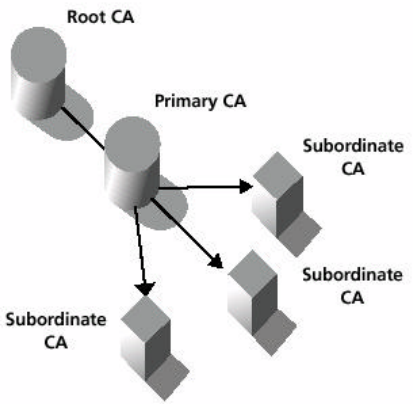


Figure 18 *Hiérarchie CA*

Il est également possible d'exploiter d'autres modèles d'architecture que le modèle hiérarchique, comme le modèle **peer to peer** (figure 19).
Les utilisateurs auront confiance dans les différentes CA de ce modèle, ce qui permet une interopérabilité efficace entre différentes PKI, pour une solution globale plus flexible.

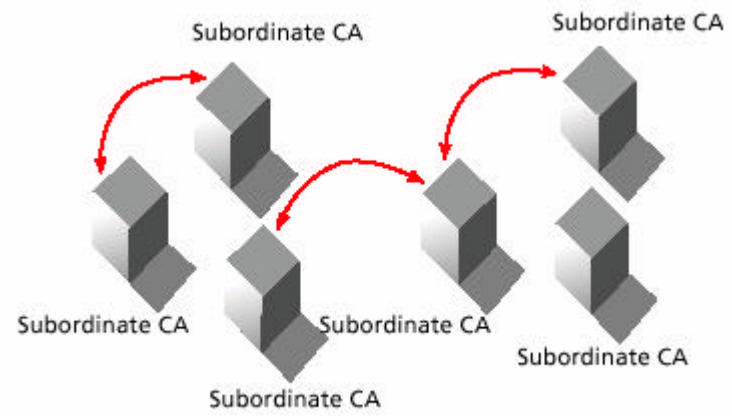


Figure 19 *Peer To Peer CA*

Keon permet également une architecture **hybride**. Il s'agit d'un mélange entre l'architecture hiérarchique et l'architecture peer to peer. Dans ce modèle, la chaîne de confiance peut être étendue à souhait.

11.6 Service de révocation

Le service de révocation doit fournir aux applications PKI-ready un moyen de contrôle sur les certificats présentés.

Le produit KEON met à disposition le service de révocation traditionnel basé sur les listes de révocation (**CRL**)

Une liste de révocation est publiée périodiquement par la CA, cette liste au format x509 contient tous les certificats révoqués. A la réception d'un certificat, les applications chargeront la dernière version de la CRL pour contrôler si le certificat n'a pas été révoqué par la CA.

Évidemment il peut exister un certain décalage entre la révocation proprement dite d'un certificat et la publication de la CRL. Ce décalage dépend de la fréquence de publication de la CRL, ce décalage peut être dévastateur pour certaines applications e-commerce.

Pour palier à ce décalage, KEON met à disposition un service plus moderne de publication basé sur le protocole **OCSP.**
L'application qui désire contrôler la validité d'un certificat présenté, interrogera le module OCSP transponder de la CA qui a émis le certificat.

Le module gérera lui même la procédure de contrôle du certificat. Le module OCSP transponder de KEON est en mesure d'interroger directement le serveur contenant les certificats pour fournir à l'application l'état de validité du certificat en temps réel.

11.7 Procédure d'enrôlement

Le produit de base **Keon CA** fournit un seul serveur physique d'enrôlement, les administrateurs se connectent au même serveur d'administration pour gérer la procédure.

Si la politique de certification choisie requiert que le client se présente physiquement à l'administrateur, il n'est pas aisé de gérer une masse de clients géographiquement dispersés.

Pour cette raison un module supplémentaire peut être ajouté au produit KEON de base, le module **KEON RA** permet d'étendre la procédure de certification en ajoutant différentes entités d'enrôlement RA physiquement reparties.

11.8 Key recovery module

Les certificats fournis par la Pki permettent aux clients d'effectuer différentes opérations cryptographiques comme la signature numérique et le chiffrage de document.
Pour éviter que des documents chiffrés par un client soient définitivement perdus si celui-ci venait à perdre sa clé privée, le produit Keon fournit un moyen de recouvrer la clé privée des clients PKI grâce au module KRM (Keon Key Recovery Module).

Ce module permet d'archiver et de recouvrer les clés privées des clients, en respectant bien évidemment les contraintes de sécurité inhérentes à cette procédure.
Ce module ajoute à la phase d'enrôlement classique une étape qui permet au client de choisir une option d'encription. Le client réceptionne ensuite un certificat spécial fourni par la PKI. Une seconde paire de clés RSA est alors générée à l'intérieur de la Pki, celle-ci est conservée sur un support physique. Le client pourra récupérer ces clés en présentant son certificat numérique.
Les clients disposeront de deux paires de clés, une paire complètement privée sera utilisée pour la signature numérique, la seconde paire sera utilisée pour le chiffrage de données. Si la clé de chiffrage est perdue, une procédure de **key recovery** pourra être entreprise par la Pki car un double de ces clés est stockée sur le support hardware.

11.9 Credential store

La technologie smart card permet de résoudre de façon efficace la problématique du stockage des clés privées, cette technologie n'est pas encore assez déployée à l'heure actuelle pour résoudre définitivement le problème, de nombreux clients conservent encore leurs clés privées à l'intérieur du browser.
Keon permet, par un module spécifique, d'offrir une alternative aux smart card. Il permet de stocker provisoirement les certificats et les clés privées des clients à l'intérieur même de la PKI. Le client, lorsqu'il a besoin de son certificat, se connecte par SSL au serveur et s'authentifie par mot de passe. Le serveur lui transmet alors son certificat et sa clé privée. Cette procédure peut s'avérer extrêmement efficace lorsque le client change régulièrement de lieu de travail et forcement de poste de travail.

Conclusion

Le produit KEON est un logiciel mur et complet, il implémente toutes les possibilités que l'on peut s'attendre à trouver dans une PKI.
Il peut tout à fait être déployé à large échelle, notamment pour fournir une autorité de certification internationale. Ses différentes méthodes d'architecture permettent de gérer les différents cas d'interopérabilité avec d'autres autorités pour définir une solution PKI globale.

Toutefois, tous ces mécanismes ne sont indispensables à nos besoins spécifiques, c'est-à-dire l'utilisation d'une PKI dans le cadre d'un VPN.

Les fonctionnalités PKI indispensables pour nos besoins sont les suivantes.

- Fournir une interface d'enrôlement sur la base d'un serveur WEB, permettant à des clients d'effectuer une demande de certificat en ligne, en remplissant un formulaire
- Fournir une interface d'administration permettant de recevoir ces demandes et de contrôler les différents champs du formulaire avant de la transmettre à un moteur cryptographique isolé du réseau.
- Signer les certificats numériques, le certificat doit être au format x509v3, les extensions ne doivent pas forcement être négociables dans le cadre d'un VPN mais le standard v3 doit être reconnu à des fins d'interopérabilité entre systèmes PKI-ready.
- Distribuer les certificats, la PKI doit fournir un mécanisme de distribution sous la forme d'un annuaire LDAP.
- Générer des listes de révocations CRL, ces listes doivent être accessibles par tous les clients PKI.

Références

[1] Security Dynamics' Keon™ Software Brings Public Key-Based Security to Mission Critical and ERP Applications
http://www.rsasecurity.com/news/pr/990118-7.html

[2] Single Sign-on SDK
http://www.rsasecurity.com/products/keon/datasheets/dskeonsso.html

[3] Certificate Authority
http://www.rsasecurity.com/products/keon/datasheets/dskeoncertificateauth.html

[4] Key Recovery Module
http://www.rsasecurity.com/products/keon/datasheets/dskeonkrm.html

[5] RSA Keon advanced PKI
http://www.rsasecurity.com/products/keon/whitepapers/advpkiwp/KEAPKI_WP_0200.pdf

[6] RSA Keon CA product overview
http://www.rsasecurity.com/products/keon/whitepapers/kca/KCATO_WP_0501.pdf

[7] Planning and Deploying a Single Sign-On Solution
http://developer.netscape.com/docs/manuals/security/SSO/sso.htm#1053955

12 Mise en œuvre d'une PKI open source pour Linux

12.1 Introduction

Les différents composants qui intéropèrent pour former une structure PKI sont souvent des équipements logiciels propriétaires, leur développement a nécessité une grande charge de travail, ce qui rend le coût d'un tel produit très onéreux.

En travaillant dans un environnement ouvert comme LINUX, il est possible de bénéficier des efforts partagés de divers groupes de travail. Ces technologies peuvent ainsi s'intégrer de manière gratuite à un projet de plus large ampleur comme une PKI.

Le prix à payer n'est donc plus d'ordre financier. Il s'agit de fournir un travail conséquent d'adaptation afin que les différents composant Open source puissent coexister dans un cadre de travail hétérogène. Les différents composants Open source à intégrer dans le logiciel proviennent de groupes de travail différents qui ont des progressions et des évolutions différentes, si l'interopérabilité entre deux modules est possible à un certain moment, aucune garantie n'est apportée sur la pérennité de cette interopérabilité.

Pour que le logiciel open PKI puisse évoluer, il est nécessaire qu'il suive la progression moyenne des différents groupes de travail rassemblés autour du vaste projet PKI.

12.2 Choix d'un projet pour une solution Open PKI

Dans le monde LINUX, il n'existe pas grands nombres de projets indépendants qui visent à implémenter une solution PKI opensource.
Deux solutions complètement différentes ont été envisagées.

- Le projet OpenCa
- Le projet Oscar

Le projet OpenCA est en réalité composé de plusieurs projets opensource indépendant.

- ***Openssl*** Ce projet fournit les méthodes de base à tout calcul cryptographie requis pour les besoins de la PKI.

- ***Apache*** Ce support réalise l'interface entre les méthodes propres à la PKI et les utilisateurs ou les administrateurs, l'interaction est réalisé par des interfaces WEB tournant sur un serveur apache.

- ***Openldap*** Ce produit met à disposition son service d'annuaire adéquat à la publication des données publiques produite par la PKI, c'est-à-dire les certificats et les CRL.

- ***OpenCA*** Ce produit fournit des méthodes sous forme de script CGI et de page HTML qui permettent de mettre en œuvre et de contrôler de façon efficace les actions de la PKI.

- ***Module Perl*** Fourni soit par différents groupes de travail indépendant, soit directement par OpenCA, ces modules serviront d'interface entre les scriptes CGI développé par OpenCA et les autres composants provenant des groupes de travail mentionné.

Ce projet est encore en plein développement, mais il est très actif dans son secteur d'activité, il contient une mailing liste efficace.

Le projet Oscar ne se base par sur OpenSSL, il fournit un système autonome basé sur du code C.
Le développement de ce produit est à l'heure actuelle terminé.

Le choix d'une solution s'est plutôt porté sur OpenCA pour deux raisons.

- Le fait que le produit Oscar ne soit plus en développement ne garanti pas qu'il contienne toutes les fonctionnalités d'une PKI. Ce produit ne peut en tous cas plus évoluer.

- La motivation première qui nous a poussée à vouloir mettre sur pied une PKI est, l'utilisation des certificats dans le cadre précis d'un VPN, or une solution d'authentification sur la base de certificat généré par OpenSSL avait déjà été testée avec succès. Le fait qu'OpenCA utilise également OpenSSL, augmente les chances d'interopérabilité entre le VPN et les certificats de la PKI.

12.3 Architecture open PKI

Une autorité de certification PKI est composée de différentes entités qui interviennent à tour de rôles dans la procédure de certification.

Ces entités sont physiquement représentées dans le cadre d'openPKI par des serveurs WEB, ces serveurs contiennent différents scripts CGI qui, combinés à des paquetages de chiffrement, permettent de mettre en œuvre tous les mécanismes de chiffrage et de contrôle nécessaires à la gestion des certificats.

La PKI est composée des différents éléments suivant la figure 20.

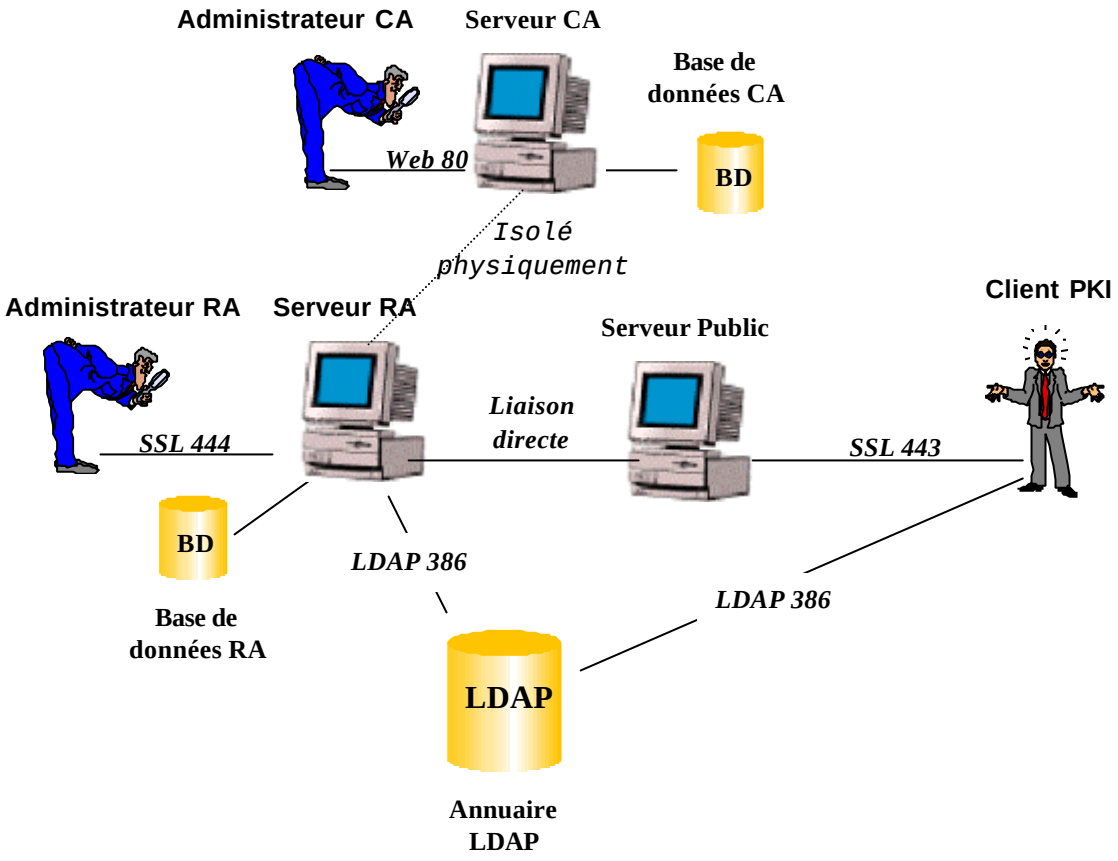


Figure 20 *Architecture open PKI*

12.3.1 Serveur Public

Cette entité est la partie visible de la PKI. Elle permet aux clients d'entrer en contact avec l'organisme PKI par l'intermédiaire d'un serveur web. Cette interface est utilisée par les clients pour transmettre des requêtes de certificats à l'autorité d'enrôlement RA, mais c'est également par cette interface que les clients recevront toutes les informations fournies par la PKI.

C'est-à-dire

- Réception du certificat numérique sollicité.
- Réception du certificat de l'autorité de certification
- Réception de la liste de révocation CRL.
- Consultation des certificats numériques de tous les clients.

Les données échangées entre le serveur public et les clients étant confidentielles, la connexion est protégée par SSL. Le serveur s'authentifie auprès de tous les clients par un certificat numérique. Étant donné que la plupart des clients n'ont pas encore de certificats lorsqu'ils accèdent à ce serveur, l'authentification client n'est pas requise.

Le serveur de la RA et le serveur publique sont installés sur le même poste.

12.3.2 Serveur RA

Ce serveur permet de gérer tout le secteur d'activité propre à la partie enrôlement de nouveaux clients. Pour cette raison, ce serveur est hautement protégé, il ne peut être accédé que par un administrateur authentifié.
L'administrateur s'authentifie par un certificat numérique spécial qui caractérise un groupe d'utilisateurs particuliers, ce groupe est composé de tous les responsables de la PKI. L'authentification se poursuit en nécessitent un login et un mot de passe administrateur. Une fois cette authentification effectuée , la connexion est sécurisée par SSL.

Le serveur RA permet aux administrateurs de la RA de stocker, de contrôler et d'approuver les requêtes émises par les clients. Toutes les requêtes sont enregistrées dans une base de données propre à la RA.
Ces requêtes seront ensuite transmises par voie sûre à l'administrateur de la CA, en vue d'une signature.

Le moyen qui à été jugé le plus sûr pour communiquer entre le serveur de la RA et le serveur de la CA est l'échange manuelle d'information, c'est à dire par disquette.

Ce serveur permet également de publier les certificats signés par la CA. La publication est faite de deux manières.

- Publication des certificats sur le serveur publique.
- Publication des certificats dans un annuaire LDAP

Le serveur de la RA et le serveur publique sont installés sur le même poste. Les échanges d'information entre ces deux entités se fait directement par fichier partagé.

12.3.4 Serveur CA

Ce serveur constitue la pièce maîtresse de l'organisme, c'est lui qui permettra de signer les requêtes de certificat approuvées par l'administrateur de la RA. Pour protéger au maximum ce mécanisme, le serveur de la CA à été volontairement isolé du réseau.

L'accès à ce serveur doit impérativement être protégé physiquement. Seul l'administrateur de la CA doit être en mesure d'accéder au poste contenant le serveur.
De ce fait, aucune sécurité informatique supplémentaire n'est ajouté à la protection physique. L'administrateur de la CA se connecte par une simple interface WEB sur le port 80, étant donné qu'il est le seul à pouvoir se connecter.

12.4 Rôles des membres PKI

12.4.1 Rôles des clients

Le rôle du client dans l'organisme se limite à une tâche de sollicitation de service. Pour effectuer une demande de certificat, celui-ci doit remplir un formulaire contenant différentes informations personnelles qui seront adjointes à son certificat. Le client génère une paire de clés RSA, dont la partie publique sera transférée avec sa demande.

Le client attend ensuite que sa demande soit traitée, avant de la récupérer par l'intermédiaire du serveur publique.

12.4.2 Rôle de l'administrateur de la RA

L'administrateur de la RA récupère les demandes fournies par les clients. Son rôle est de contrôler ces requêtes. Il a le droit de veto sur ces requêtes, c'est-à-dire qu'il peut rejeter toute requête qui ne correspond pas à la politique de certification de l'organisme.

La politique de certification est la suivante.
Les demandes qui ne contiennent pas de clé associée seront systématiquement rejetées. Après une sollicitation de certificat, le client devra se présenter physiquement auprès d'un administrateur RA, l'administrateur approuvera la requête uniquement lorsqu'il aura contrôlé l'identité du client par le truchement d'une pièce d'identité.

Lorsque la demande de certificat à été approuvée par l'administrateur, celui-ci signe le document. La requête signée par l'administrateur compose un standard de requête au format PKCS#10 qui peut être transférée à la CA.
Le format PKCS#10 est un standard pour le format de requête

ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-10/pkcs-10v1_7.pdf

L'administrateur de la RA a également une tâche de publication, c'est lui qui est responsable de récupérer les certificats et la CRL signée, puis de les rendre accessibles pour les clients. Cette publication dans le serveur publique se fait de manière automatique lors qu'il reçoit les certificats et la CRL de la CA, mais il est nécessaire d'ajouter ces informations dans l'annuaire LDAP.

12.4.3 Rôle de l'administrateur de la CA

Cet administrateur est hiérarchiquement le plus élevé de tout l'organisme. Sa responsabilité est plus grande que celle de l'administrateur de la RA car c'est lui qui signera le certificat proprement dit.

Sa tâche consiste à réceptionner les requêtes de certificats au format PKCS#10, puis à vérifier la signature de l'administrateur qui a émis cette requête.

La signature sur la requête PKCS#10 détermine de façon univoque l'identité de l'administrateur RA qui a contrôlé la requête. L'identité est représentée par le numéro de série du certificat administrateur.

L'administrateur de la CA ne contrôle en principe pas la validité des données contenue dans le requête. Il fait confiance dans le travail de contrôle effectué par l'administrateur de la RA.

Il validera ensuite la requête en l'a signant à l'aide de la clé privée du certificat root CA, le certificat ainsi formé sera conforme au standard de certificat X509. Le certificat peut être retourner à l'administrateur de la RA.

L'administrateur de la CA peut également révoquer des certificats qui ne sont plus conforme à la politique de certification de l'organisme.
Toute la liste des certificats révoqués doit être publiée dans une CRL. Cette liste comme tous certificat numérique conforme au standard x509 a une validité limitée, par exemple 24h. Comme la plupart des applications PKI-ready, notamment le VPN contrôle cette liste avant d'accepter la validité d'un certificat numérique, il est nécessaire de fournie une nouvelle CRL périodiquement.

L'administrateur doit systématiquement générer une nouvelle CRL, quand bien même aucun nouveau certificat n'a été révoqué.
Cette contrainte est indispensable pour permettre une interopérabilité avec les applications PKI-ready.

12.5 Zone d'enrôlement

Suivant le déploiement de la PKI, le nombre de requêtes des clients peut être relativement élevées, provoquant une surcharge de travail de la part de l'administrateur de la RA. Pour

diviser le travail de contrôle de l'administrateur de la RA, l'administration de la RA est effectuée par plusieurs administrateurs simultanément.
A cet effet, trois zones d'enrôlement ont été définies, chaque zone étant gérée par un administrateur RA. Le client, lors de sa demande de certificat, choisit la zone d'enrôlement qui lui convient.
Cette division permet de repartir géographiquement les lieux d'enrôlement, tout en garantissant une bonne sécurité dans la procédure de contrôle.

Cette politique d'enrôlement a pour avantage de repartir les administrateurs géographiquement, mais tous les administrateur se connectent au même serveur RA. Ce mécanisme n'est donc pas aussi puissant que celui proposé par le produit KEON (Keon RA).

12.6 Hiérarchie de CA

L'implémentation de la Pki basée sur OpenCA ne permet pas à l'heure actuelle de déployer une hiérarchie de CA comme le permettait le produit KEON.
La possibilité de hiérarchisation est toutefois possible, une fois la PKI mise sur pied, il est possible de créer une seconde CA comme CA subordonnée, un certificat numérique signé par la CA root peut lui être attribué, la CA subordonnée sera en mesure de signer des certificat.

Toutefois la procédure n'est pas aussi évidente que celle proposée par KEON.
Il n'est en revanche pas possible de mettre en œuvre une certification croisée dans le style peer to peer, encore moins une architecture hybride.

12.7 Protection des clés privées

Avec le système Keon, le client pouvait spécifier au moment de l'enrôlement, le support sur lequel il désirait conserver ses clés et ses certificats. Le produit openCa ne fournit pas de mécanisme similaire permettant un stockage systématique des certificats et des clés privées sur support hardware.

Etant donné que la protection des clés privées est absolument nécessaire pour assurer une solution sécurisée complète, un mécanisme alternatif peut être mis en œuvre.

Si la procédure d'intégration du certificat dans le support hardware ne peut être faite automatiquement par les scriptes d'OpenCA, il est tout à fait possible de réaliser cette procédure de façon manuelle, c'est-à-dire en exportant le certificat et la clé privée associée dans la smart card depuis les propriétés de sécurité du browser.

Sur un système NT, les supports cryptographiques hardware comme la smart card sont accessibles depuis le browser par l'intermédiaire d'un driver spécifique.

Le groupe de travail MUSCLE met à disposition des drivers pour permettre une utilisation de smart card dans le monde LINUX, il est ainsi possible d'offrir une protection des clés privées quel que soit le système d'exploitation utilisé.

http://www.linuxnet.com

12.8 Key recovery

Le produit OpenCa ne fournit aucun mécanisme capable d'effectuer une telle procédure, d'après les développeurs, ces fonctionnalités ne devraient pas être implémentées avant 2003.

12.9 Liste de révocation

Comme il a été dit précédemment, openCa utilise la manière traditionnelle pour communiquer la liste des certificats révoqués.
Les applications sont responsables de charger elles-mêmes la CRL, soit depuis le serveur publique, soit directement depuis l'annuaire LDAP.

Une solution basée sur un transponder OCSP devrait prochainement être intégrée dans le produit OpenCA.(d'après OpenCA team)

12.10 Interopérabilité

Les certificats fourni par la Pki OpenCa sont parfaitement conformes au standard ASN1, ces certificats permettent donc une interopérabilité entre différents composants PKI-ready, notamment les VPN basés sur Freeswan.

En ce qui concerne l'interopérabilité entre plusieurs PKI, l'interopérabilité est moins évidente qu'avec le produit KEON.
Il est possible d'exploiter le modèle hiérarchique en créant une seconde génération de CA dont le certificat est signé par le CA root, mais l'architecture peer to peer n'est pas possible. La possibilité d'utiliser un certificat root qui n'est pas auto signé est tout à fait possible, ce qui permettrait d'intégrer deux PKI différente uniquement avec une architecture hiérarchique.
.

13 PKI open source avec OpenCA

13.1 Projet OpenCa

Le projet OpenCA étant en plein développement, les versions disponibles sur le site du groupe (www.openca.org) sont souvent instables et non complètes, un grand nombre de modules Perl supplémentaires et de patchs sont également disponibles pour corriger les erreurs des versions précédentes.

Dans une première installation, la version d'OpenCA utilisée était la version 0.2.0-5, il s'agissait de la dernière version stable connue du paquetage.

Cette version a été extrêmement fastidieuse à installer, elle contenait de nombreux bugs et d'incompatibilité qui rendait cette solution insatisfaisante.

Cette version présentait de graves incompatibilités avec OpenLDAP et une maturité précaire dans les scripts proposés ce qui a poussé les développeurs à arrêter le développement de cette version et à repartir sur des bases différentes.

Une seconde version stable de OpenCa est apparue simultanément avec la fin de l'installation de OpenCa0.2.0-5, il s'agit de la version 0.8.0.

Cette version est complètement différente de la version 0.2.0, la structure du paquetage est améliorée par de nombreux nouveaux scripts.
Son installation est fortement simplifiée par une procédure d'auto configuration et par un script d'installation automatique des modules perles.

Pour cette raison, ce document ne contient que la procédure permettant de mettre en œuvre une PKI open source basée sur a version 0.8.0 de OpenCA. Toutefois, une notice d'installation de OpenCA 0.2.0-5 est disponible en annexe sur CD.

OpenCA étant la pièce directrice de tout le projet, les versions des autres composants doivent être scrupuleusement adaptées, pour cette raison la configuration utilisée avec OpenCA lors de sa mise en œuvre a utilisé les modules suivants :

Configuration utilisée pour OpenCA

- Mandrake 8.0 + suse 7.2
- Openssl-snap-20011108
- Perl(5.6.0)
- Mod_ssl(for Apache)
- OpenLdap 2.0.18
- Apache 1.3.19

La mise en œuvre de la PKI se décompose en plusieurs étapes

- Installation des préliminaire sur le poste de la CA
- Installation des préliminaire sur le poste de la RA
- Installation d'OpenCa partie CA
- Installation d'OpenCA partie RA et publique
- Configuration serveur CA
- Initialisation CA
- Configuration serveur RA et serveur publique
- Interopérabilité CA et RA
- Initialisation RA

13.2 Préliminaire pour OpenCa 0.8.0

13.2.1 OpenSSL

Le projet OpenSSL met à disposition dans son paquetage grand nombre de fonctionnalités cryptographique et d'outils pour implémenter une sécurité robuste sur SSL et TLS.

Les fonctions de base mises en œuvre par openSSL

- Création de paire de clés RSA, paramètres Diffie-hellmann et DSA
- Signature numérique au format PKCS#7
- Création d'une requête de certificat au format PKCS#10
- Fonction de hachage
- Création de certificat X509, et CRLs
- SSL/TLS client et serveur
- Signature de mail S/MIME
- Vérification de certificat X509
- Divers outils de conversion de format

Les besoins spécifiques de la PKI mis en œuvre par OpenCA ont poussé les développeurs de OpenSSL à ajouter des fonctionnalités supplémentaires au paquetage.

Par exemple la possibilité d'ajouter des extensions au certificat X509, ou de fournir des outils de vérification des signatures numériques des certificats, sont requit pour une PKI.
Pour cette raison, la version de OpenSSL à utiliser doit être en mesure de traiter ces exigences particulières, sinon un patch doit être appliqué pour mettre à jour la version.

Mais actuellement OpenSSL inclus dans ces dernières version instables de son paquetage (snapshot) les extensions nécessaires à son interopérabilité avec OpenCA.

Le paquetage OpenSSL doit être installé aussi bien sur le poste de la CA que sur le poste de la RA.

Tar vxfz OpenSSL{version}.tar.gz **Cd OpenSSL{version}**	Décompression
./configure	Configuration
make **make test**	Compilation
make install	Installation

13.2.2 Installation d'un interpréteur Perl.

Les scripts CGI mis à disposition par OpenCA utilisent le langage Perl, pour cette raison il est nécessaire de disposer d'un interpréteur à jour.

En principe l'interpréteur est incorporé au système lors de l'installation de la Mandrake ou de la Suse.

Plusieurs versions de l'interpréteur ont été testées.

- Perl 5.6.0
- Perl 5.6.1

Dans les deux cas, les résultats ont été satisfaisant, les scripts CGI ont pu être exécuté. Mais dans tous les cas, la version de l'interpréteur ne doit pas être inférieur à 5.6.0.

Perl est un paquetage qu'il est possible d'obtenir sous forme RPM, il doit être installé sur les deux postes

http://rpmfind.net

13.2.3 Installation script perl et modules spécifiques

Pour exécuter les scriptes CGI aussi bien depuis le poste de la CA que depuis la RA il est nécessaire d'installer le module perle `CGI.pm`

Tar vxfz CGI.pm-2.78.tar.gz **Cd CGI.pm-2.78**	Décompression
Perl makefile.PL	Construction d'un make file

Make Compilation
Make test
Make install Installation

le paquetages `expat` est également requit sur les deux postes.

Rpm Uvh expat-1.95.2-1.i686.rpm

13.2.4 Installation OpenLdap sur le poste de la RA

Bien que la RA puisse être installée sans utiliser openldap, il est indispensable de mettre en œuvre ce système efficace de publication.

openldap 2.0.18 peut être installée de plusieurs façons.

De manière systématique

Rpm Uvh openldap-2.0.18-1mdk.i586.rpm
Rpm Uvh openldap-clients-2.0.18-1.mdk.i586.rpm
Rpm Uvh openldap-migration-2.0.18-1.mdk.i586.rpm
Rpm Uvh openldap-serveur-2.0.18-1.mdk.i586.rpm

Ou de façon automatique à l'aide de gestionnaire de paquetage de la mandrake
et des sites de mise à jour `rpmfind.`

13.3 Installation de OpenCa 0.8.0 (Partie CA)

Le paquetage OpenCA doit être installé sur les deux postes, en commencent par le poste de la CA.

Tar vxfz OpenCA-0.8.0.tar.gz Décompression
Cd OpenCA-0.8.0

13.3.1 Pré configuration OpenCa

OpenCA 0.8.0 fournit un script de configuration du paquetage qui facilite grandement son installation.

./config –with-user=wwwrun –with-group=nogroup –with-base-url=10.192.73.28 –with-org=tcomCA –with-country=ch –with-loc=yverdon –with-ldap-url=10.192.73.28

Les paramètres sont les suivants

`--with-user=`	Nom d'utilisateur du serveur apache
`--with-goup=`	Nom de groupe du serveur
`--with-base-url=`	Adresse de distribution de base, il s'agit de l'url du serveur public
`--with-org=`	Le nom de l'organisation formera avec l'attribut «country » la « basedn » pour l'annuaire LDAP.
`--with-loc=`	Localité de la CA
`--with-ldap-url=`	Url de l'annuaire ldap, l'annuaire est placé sur le même poste que le serveur de la RA.

Le nom et le groupe du serveur apache sont des informations qui sont disponibles dans le fichier de configuration du serveur.
`/httpd/httpd.conf`
Ou `/etc/httpd/commonhttpd.conf`

13.3.2 Installation de la CA

Tous les composants propres à la CA sont installés par la commande suivante :

`Make full-ca`

Cette commande crée les répertoires propres à la CA, les pages HTML et les scripts CGI de la CA ; mais les modules perles nécessaires au fonctionnement du serveur de la CA sont également automatiquement installés.

Il s'agit des modules suivants

```
Convert-ASN1
URI
XML-Parser
Digest-MD5
MIME-Based64
IO-Socket-SSL
Perl-ldap
OpenCA-CRL
OpenCA-CRR
OpenCA-Configuration
OpenCA-DB
OpenCA-DBI
```

Configuration 1 *Module Perl de la CA*

On constate que le script installe tous les modules, y compris le modules pour l'interaction avec LDAP qui est inutile pour la CA.

Le répertoire et les fichiers propres à la CA sont installés dans le répertoire
`/usr/local/OpenCA`

Les scripts CGI et les pages HTML utilisées par le serveur Apache sont installé dans le répertoire.
`/home/httpd/`

Le fichier d'auto configuration n'est pas parfait, il est donc nécessaire d'apporter quelques modifications au fichier de configuration de la CA.

`/home/httpd/cgi-ca/conf/ca.conf`

Il s'agit de vérifier dans la section `crypto` , que l'emplacement de l'exécutable `openssl` corresponde bien à la dernière version d'openssl. Et non pas la version installée par défaut lors de l'installation de Linux. Le fichier de configuration «ca.conf » réalisé, comme tous les fichier de configuration sont disponible en annexe sur le CD :

```
##Crypto Section
openssl  « /usr/local/ssl/bin/openssl »
```

Configuration 2 *Lien sur Openssl (extrait ca.conf)*

Malgré ce changement, il a été constaté que nombre de scripts recherchaient encore l'exécutable au mauvaise emplacement.

Les scripts ont malheureusement tous été configurés pour rechercher l'exécutable `openssl` depuis l'emplacement `/usr/bin/openssl`. Pour s'assurer que la CA n'utilise pas l'ancienne version de openssl, source d'erreur classique, il a été nécessaire d'effectuer un lien symbolique sur ce fichier.

Pour effectuer ce lien symbolique, la procédure est la suivante :
Par mesure de sécurité, le fichier est archivé

`mv /usr/bin/openssl  /usr/bin/openssl_back`

Puis la redirection est effectuée

`ln –s /usr/bin/openssl  /usr/local/ssl/bin/openssl`

Cette redirection permet de s'assurer que dans tous les cas la dernière version de openssl sera utilisée, évitant ainsi des erreurs incompréhensibles par la suite.

Cette procédure devra être ré effectuée sur le poste de la RA-

A ce stade, la partie CA de la PKI est parfaitement installée. Il s'agit par la suite de configurer le serveur apache qui hébergera l'interface WEB de la CA.

13.4 Installation de OpenCa 0.8.0 (Partie RA et interface publique)

La RA et le l'interface publique sont installée sur un deuxième poste qui est connecté au réseau. Sur ce poste la distribution de Linux utilisée est une Mandrake 8.0

Le paquetage d'openCa doit être décompresser sur le nouveau poste.

```
Tar vxfz OpenCA-0.8.0.tar.gz
Cd OpenCA-0.8.0
```

13.4.1 Pré configuration OpenCa

Avant d'exécuter le script de configuration de openca, il est souhaitable de modifier certains fichiers de configuration pour nos propres besoins.

Le fichier de configuration `raserver.conf` doit être édité.

/OpenCa-0.8.0/src/cgi-bin/cgi-raserver/conf/raserver.conf

Dans la section consacrée au mail, les paramètres `Mailsendername, mailsenderaddress` doivent être définis de manière à correspondre au nom et au mail de l'administrateur de la RA.

Pour permettre une flexibilité accrue dans la gestion de l'enrôlement, la zone d'activité de la RA a été divisée en trois. Ainsi il est possible de définir un administrateur par zone.

Les trois zones d'enrôlement sont les suivantes :

- Systèmes
- Déploiement
- Développeurs

Dans le fichier de configuration `raserver.conf,` dans la section générale.

Le paramètre `RAChoiseBaseSheet` doit été divisé en trois zones,

RAChoiseBaseSheet, Systems, Delpoyment, developers

Pour donner une certaine coloration aux certificats, les clients sont également divisé en trois catégories d'utilisateurs. Ces catégories seront utilisées pour diviser les utilisateurs dans l'annuaire LDAP, d'après l'attribut ou (organization Unit)

- Utilisateur
- Développeur
- Employer

Pour permettre d'utiliser ces trois catégories, il est nécessaire de modifier le fichier de configuration de l'interface publique.

`/OpenCa-0.8.0/src/cgi-bin/cgi-public/conf/public.conf`

Dans la section générale, le paramètre « Organization Unit » doit définir les différentes catégories d'utilisateur

```
Organization Unit
TcomCa user, TcomCa develloper, TcomCa employer

#Seulement un niveau d'unité est nécessaire
OrganizationUnitNumber 1
```

Configuration 3 *Définition des groupes d'utilisateurs (extrait public.conf)*

Il est également nécessaire de spécifier le degrés hiérarchique des unités. Il y a possibilité de définir des sous unités.

Le script de configuration peut être exécuté en utilisant les différents paramètres, comme pour l'installation de la CA.

```
./config –with-user=apache –with-group=apache –with-base-url=10.192.73.28 –with-org=tcomCA –with-country=ch –with-loc=yverdon –with-ldap-url=10.192.73.28
```

Le nom et le groupe du serveur apache ne sont pas les même suivant les distribution de linux. Voire `/httpd.conf` ou `commonhttpd.conf`

13.4.2 Installation Ra et interface publique

Cette commande permet d'installer les répertoires de la RA ainsi que les scripts et les pages html pour le serveur RA, mais également les pages html de l'interface publique.

`Make full-raserver`

Les fichier suivants sont installés :

La base de données propre à la RA est installée dans le répertoire
`/usr/local/RAServer`

Les scripts CGI et les pages HTML utilisées par le serveur Apache
`/home/httpd/`

Média d'échange

La RA doit communiquer de maniérée sûre avec la CA, c'est à dire par disquette. Il est donc nécessaire de modifier le fichier de configuration de la RA pour rendre possible ce moyen d'échange.

Dans le fichier de configuration de la RA
`/home/httpd/cgi-raserver/conf/raserver.conf`

Dans la section `In/Out`

Modifier les répertoires d'échange entre la CA et la RA

```
#In/Out Section
# Les répertoires d'échanges entre la CA et la RA sont modifier pour
#utiliser le périphérique floppy.
ExportDev  « /mnt/floppy/openca-outca.tar
ImportDev  « /mnt/floppy/openca-inca.tar
```

Configuration 4 *Interface RA/CA (extrait raserver.conf)*

Vérifier également que le chemin vers l'exécutable de openSSL soit bien définit

Dans la section «Crypto Section»
Vérifier que le chemin vers openssl soit correct

```
openssl  « /usr/local/ssl/bin/openssl »
```

13.5 Apache

Pour mettre en œuvre une PKI dont les éléments sont accessible par le WEB, l'utilisation d'un serveur est indispensable. Mandrake et Suse comme la plupart des distributions de linux fournissent le serveur WEB apache lors de l'installation de Linux.

Le serveur apache de base ne comporte toutefois pas toutes les fonctionnalité de sécurité nécessaires aux besoins de la PKI.

Les clients qui effectuent une demande de certificat par l'intermédiaire du serveur public ont besoin de confidentialité sur leurs données. De plus ils doivent être sur de l'identité réel du serveur auquel il sont connectés, évitant ainsi l'attaque du « men in the middle ».

Cette confidentialité et cette authentification sont fournies par l'ajout du module `mod_ssl` for Apache.

13.5.1 mode SSL

Ce module fournit un chiffrage puissant pour apache en utilisant le protocole SSL (Secure Socket Layer) mais également son successeur TLS (Transport Layer Security).

L'authentification est réalisée par l'utilisation de certificats numériques, deux cas sont prévu.

- Authentification serveur : Le serveur transmet toujours sont certificat en début de négociation.
- Authentification client : Certaines applications exigent que les clients soient en mesure de fournir un certificat avant de pouvoir se connecter.

Le serveur contenant la CA est isolé physiquement du réseau. Les risques d'intrusion sont donc écartés de ce point de vue. Le serveur de la CA n'a pas besoin d'être protégé par SSL.

Mais le paquetage mod_ssl pour apache doit être installé sur le poste de la RA. Avec la distribution Mandrake de Linux, mod_ssl peut être installé par défaut avec le serveur Apache. Pour les autres distributions, mod ssl peut être obtenu depuis le site suivant :

http://www.modssl.org

13.5.2 Configuration du serveur apache

Sur la Mandrake, le fichier de configuration de apache est composé d'un fichier principal et de plusieurs fichiers de configurations liés.

Tous les fichiers de configuration du serveur apache sont situé dans le répertoire suivant.

`/etc/httpd/conf/`

Le fichier de configuration du serveur apache est en réalité divisé en plusieurs fichiers.

```
Httpd.conf
Commonhttpd.conf
Mod_ssl.conf
```

Le fichier `httpd.conf` est le fichier de base de configuration. En principe il n'est pas nécessaire de modifier son contenu. Ce fichier contient des références sur d'autres configurations qui elles, doivent être configurées suivant les besoins, ce sont entre autre les fichier `commonhttpd.conf` et `mod_ssl.conf.`

Le fichier `commonhttpd.conf` contient tous les paramètres qui permettent d'adapter le serveur à nos besoins. Il contiendra tous les paramètres de configuration du serveur de la CA.

Le fichier `mod _ssl.conf` permet d'ajouter des paramètres de configuration particulière à un serveur sécurisé. Ce fichier contiendra la configuration de base pour les serveurs de la RA et de l'interface publique.

13.5.3 Configuration serveur de la CA

Le serveur de la CA est un simple serveur WEB, travaillant sur le port 80. Il s'agit donc d'apporter les modifications nécessaires au fichier `commonhttpd.conf`, pour intégrer les pages HTML et les script CGI de la CA aux fichiers standard de configuration.

Le fichier de configuration est disponible dans sa totalité en annexe.

Les points principaux à modifier dans le fichier `commonhttpd.conf` sont :

- Redéfinir le « document root » avec le répertoire des pages HTML de la CA, c'est-à-dire

  Documentroot **`/home/httpd/htdocs-ca.`**

- Définir l'emplacement des scripts CGI de la CA.

  Scriptalias /cgi-bin/ **`/home/httpd/cgi-ca.`**

- Restreindre l'accès sur les répertoires contenant les pages HTML et les scripts de la CA.

```
 #Limiter toutes modifications sur les répertoires et leurs contenus.
<Directory « /home/httpd/htdocs-ca/ »>
   AllowsOverride None
   Options FollowSymLinks
   Order allows,deny
   Allows from all
</Directory>

<Directory « /home/httpd/cgi-ca/ »>
    AllowsOverride None
    Options FollowSymLinks
    Order allows,deny
    Allows from all
 </Directory>
```

Configuration 5 *Droit d'accès sur les répertoires de la CA (extrait commonhttpd.conf)*

Note : Le fichier de configuration de la CA, comme tous les fichiers de configuration qui seront réalisé, sont disponible dans leur totalité sur le CD en annexe.

Une fois le serveur configuré spécifiquement pour la CA, il est nécessaire de redémarrer le serveur.

```
/etc/init.d/httpd restart
```

Si tout à été configuré correctement, la page d'accueil de la CA est disponibles en utilisant le navigateur **Netscape** avec comme URL l'adresse IP du poste ou l'adresse de loop-back well known 127.0.0.0.

13.5.4 Initialisation de la CA

Une fois connecté sur le serveur de la CA, la tâche de l'administrateur consiste à initialiser la base de données de la CA, puis à générer une paire de clés RSA et le premier certificat de la PKI. Il s'agit du certificat root CA, le seul certificat auto-signé de toute la hiérarchie.

- Initialisation de la base de données

Dans le menu `Initialisation`

Initialize Database

Cette procédure a pour effet de vider la base de données, cette procédure ne doit être effectuée qu'une seule fois lors de l'initialisation

- Puis créer la paire de clés RSA :

Generate new CA secret key

La clé privée est stockée dans le répertoire `/private.` de la CA.

- Créer une requête de certificat .

L'administrateur peut ensuite effectuer une requête de certificat. Dans cette étape, plusieurs informations personnelles seront demandées. Ces informations constitueront le DN (distinguish name) du certificat root CA. Ce « Dn » deviendra l'entrée de base dans l'annuaire LDAP, pour cette raison il est préférable de s'informer du formalisme autorisé auprès de l'administrateur de l'annuaire. Sans quoi toute la hiérarchie PKI ne pourra pas être intéropérable avec un annuaire LDAP existant.

Generate new Ca certificate request

- Signer le certificat de la CA :

La procédure de signature requiert le mot de passe administrateur CA, celui qui a été introduit lors de la génération des clés RSA de l'administrateur.

Generate self signed CA Certificate

- Créer la chaîne de confiance

La nouvelle version de OpenCA permet de générer une chaîne de certificat, il s'agit de la chaîne de confiance comportant tous les certificats depuis le clients PKI jusqu'au certificat root de la CA.

Rebuild CA chain

Media d'échange

La CA a volontairement été isolée du réseau pour des questions de sécurité. Le media de transport qui a été choisi pour permettre des échanges entre la RA et la CA est la disquette.

Ce moyen de transfert est le plus sûr, il permet de contrôler manuellement les échanges avec la CA.

L'étape suivante consiste donc à configurer la CA pour qu'elle utilise ce media pour stocker les messages avec la RA.

Dans le fichier de configuration de la ca.

`/home/httpd/cgi-ca/conf/ca.conf`

Dans la section consacrée aux entrées sorties

```
#In/Out Section
# Les répertoires d'échanges entre la CA et la RA sont modifier pour
#utiliser le périphérique floppy.
ExportDev  « /media/floppy/openca-outca.tar
ImportDev  « /media/floppy/openca-inca.tar
```

Configuration 6 *Interface CA/RA (extrait ca.conf)*

La distribution de linux nécessite de monter et de démonter le lecteur de disquettes à chaque utilisation.

Depuis la console, les commandes suivantes permettent d'utiliser correctement le lecteur de disquette depuis le serveur apache.

Pour monter le lecteur de disquette, sur une SUSE 7.2

mount /dev/fd0 /media/floppy –o uid=wwwrun

Le lecteur « floppy » doit être accessible depuis le serveur apache aussi bien en écriture qu'en lecture.

Pour démonter

umount /media/floppy

Cette procédure de montage, démontage devra être effectuée chaque fois que des informations seront échangées entre la CA et la RA.

13.5.5 Configuration serveur de la RA

Il s'agit de configurer le serveur apache pour héberger le serveur de la RA sur SSL. Le serveur est configuré avec les contraintes suivantes :

- La configuration du mode SSL doit imposer l'authentification serveur, pour cela un certificat numérique doit être généré à l'intention du serveur public.
- Le serveur ne doit être accédé que par des administrateurs autorisés. De ce fait, il doit être configuré pour exiger l'authentification client

La configuration du serveur apache pour la RA se décompose en plusieurs étapes qu'il est absolument nécessaire d'effectuer dans l'ordre.

1) Créer un fichier de configuration raSSL.conf basé sur le fichier exemple de SSL `ssl.default-vhost.conf`. Il doit permettre de se connecter au serveur sur le port 444 en utilisant les clés est les certificats fournit pas SSL. L'authentification client ne doit pas encore être requise pendant la phase de configuration.
2) Exporter le certificat root CA et la CRL sur le poste de la RA
3) Générer un certificat serveur à l'aide d'un scripte fournit par OpenCA
4) Générer un certificat administrateur au format exportable PKCS#12
5) Importer le certificat administrateur dans le browser Netscape de la RA
6) Modifier le fichier de configuration du serveur Apache raSSL.conf afin d'utiliser les clés et le certificat créés à son intention.

1. Configuration en utilisant les certificats par défaut

Pratiquement, le serveur est configuré en premier lieu en utilisant les clés et un certificat exemple (dummy) fournit par mod ssl. Une fois la connexion possible avec ce serveur, le certificat et les clés de SSL seront remplacée par un certificat numérique produit spécialement par notre propre PKI.

Le serveur apache permet de mettre en œuvre différentes configurations appelées « Virtual host ». Cette possibilité fournie par apache, permet de définir différents serveurs avec des configurations spécifiques sur le même poste, chaque serveur est différencié par un port TCP de valeur différente.

Pour le serveur de la RA, le port TCP utilisé est le 444.

Un fichier de configuration nommé `raSSL.conf` est édité, il se base sur le fichier exemple de mode ssl, `ssl.default-vhost.conf`.

Il s'agit de modifier dans ce fichier les informations suivantes.

```
# Définition d'un virtual host sur le port 444
VirtualHost_default- :444

#Emplacement des pages html et des scripts CGI du serveur RA
DocumentRoot /home/httpd/htdocs-raserver
ScriptAlias /cgi-bin/ /home/httpd/cgi-raserver
#Journal d'erreur du serveur, utilisé pour debuger
ErrorLog /var/log/http/raserver-error_log

#L'accès à ces emplacements doit être restreint
<Directory «/home/httpd/htdocs-raserver»>
  AllowOverride None
  Option None
  Order allow,deny
  Allow from all
</Directory>
<Directory «home/httpd/cgi-raserver »>
```

```
  AllowOverride None
  Option None
  Order allow,deny
  Allow from all
</Directory>
```

Configuration 7 *Virtual host serveur RA(extrait raSSL.conf)*

Les paramètres spécifiques au serveur SSL suivant, doivent utiliser les certificats et les clés fournies par mod SSL.

```
#Certificat SSL du serveur public
SSLCertificateFile conf/ssl/sslserver.pem
#Clé privée SSL du serveur public
SSLCertificateKeyFile conf/ssl/sslserver_key.pem
#certificat de l'autorité qui à signé le certificat SSL
SSLCACertificateFile  /etc/httpd/conf/ssl/cacert.pem

#L'authentification client ne doit pas encore être requise
SSLVerifyClient none
```

Configuration 8 *Paramètres SSL serveur RA (extrait raSSL.conf)*

Étant donné que le port utilisé pour le serveur de la RA n'est pas le port «well known » https, il est nécessaire de configurer le fichier `mod_ssl.conf` afin qu'il considère le port 444 comme port SSL à part entière.

```
## SSL Support
## When we also provide SSL we have to listen to the
## standard HTTP port (see above) and to the HTTPS port
##
Listen 443
Listen 444
```

Configuration 9 *Ajout du port SSL 444 (extrait mod_ssl.conf)*

Le fichier de configuration `raSSL.conf` est ensuite ajouté à la liste des fichiers de configuration dans le fichier `httpd.conf (Configuration 10)`

```
Include conf/ssl/ raSSL.conf
```

Configuration 10 *Ajout de la configuration raSSL.conf (extrait httpd.conf)*

Il s'agit uniquement de pouvoir se connecter au serveur, mais rien ne doit être entrepris pour le moment.
Pour se connecter au serveur de la RA

https://10.192.73.28:444

2. **Importer le certificat root CA et la CRL**

Toute la hiérarchie Pki repose sur une chaîne de confiance, le premier maillon de la chaîne est le certificat root de la CA. Celui-ci doit être publié afin que tous les clients PKI puissent vérifier la signature des certificats échangés.

La première étape consiste à exporter le certificat root et la CRL depuis le serveur de la CA sur le média de transport (disquette).

Depuis le serveur de la CA, dans le menu `Management`

Choisir ***`Export CA certificate.`***

Le certificat est transféré sur la disquette, au format pem et der.
Il s'agit ensuite de transmettre la première CRL, cette liste est en principe vide mais elle est indispensable à la procédure de contrôle du serveur apache

Dans le menu `CRL`

Choisir ***`Issue new CRL`***

Cette liste est semblable aux certificats utilisateurs, elle nécessite une signature de l'administrateur de la CA.

Dans le menu ***Input and Output***

Choisir ***`Export CRL`***

Le certificat root ainsi que la CRL peuvent être récupéré par l'administrateur de la RA.

En se connectant au serveur de la RA par l'URL suivant

https://10.192.73.28 :444

Puis une fois sur le site de la RA
Dans le menu `Administration`
Choisir ***RAServer***

Puis ***Initialise Database***
Import CA certificate
Rebuild Ca chain.

Le certificat root est automatiquement introduit dans le répertoire suivants au format PEM et DER.

`/usr/local/RAServer/cacert.pem`

Il s'agit ensuite d'importer la CRL.

Dans le menu `Inbound`
Choisir ***ImportCRL***

La CRL est automatiquement importée au format PEM et DER, l'emplacement de la CRL au format PEM est le suivant.

/`usr/local/RAServer/crl/cacrl.pem`

3. **Créer un certificat pour le serveur de la RA**

Le serveur de la RA utilise pour le moment les certificats exemples fournit par mod ssl. Dans cette étape, il s'agit de créer un certificat serveur pour le serveur de la RA.
Étant donné que la PKI n'est pas encore opérationnelle, il n'est pas possible de fournir de certificat par la voie traditionnelle.

OpenCA fournit un script qui permet d'attribuer des certificats de façon directe. Ce script ne doit jamais être utilisé pour attribuer des certificats clients, car aucun contrôle n'est effectué lors de cette procédure.

Le script **openca-newcert** est disponible depuis le poste de la CA dans le répertoire `scripts` de OpenCA

`/OpenCA-0.8.0/scripts/`

Le script **openca-newcert** permettant de générer des certificats numériques, conduit à des erreurs lors de la génération des certificats, l'extension «commentaire » du certificat ne peut être faite correctement.

Il s'est avéré qu'il s'agissait d'une erreur des développeurs du scripte. L'erreur peut être corrigée en mettant en commentaire la ligne suivante.

```
# nsComment  =$ENV: :nsComment.
```

Dans les fichiers de configurations.

`/usr/local/OpenCA/conf/openssl/extfiles/**Server_Certificate.ext**`
`/usr/local/OpenCA/conf/openssl/extfiles/**User_Certificate.ext**`

Puis le script suivant peut être utilisé, il permet de créer un certificat numérique au format PEM, aussi bien pour les serveurs que les administrateurs.

Sh openca-newcert

Note : La différence fondamentale entre un certificat serveur et un certificat client, d'après OpenCa, est qu'un serveur ne peut en principe pas signer des documents. La signature est une procédure volontaire, elle ne peut donc pas être faite de manière automatique par un serveur, d'où une limitation dans l'extension du certificat serveur.

Lors de l'introduction des données dans la requête du certificat, il est important d'introduire des données conformes au standard de l'organisation, afin que le certificat puisse être introduit dans l'annuaire LDAP, notamment le CN doit correspondre au nom du serveur.

Le certificat serveur au format PEM est stocké provisoirement dans le répertoire.

`/usr/local/OpenCA/outbound.`

La clé privée du serveur est disponible dans le répertoire

`/usr/local/OpenCA/private`

Un double de ces clés doit toujours être présent dans la base de données de la CA. Cette politique a été définie afin d'être en mesure de réagire en cas de perte du certificat serveur. Il s'agit du seul cas de key recovery possible de toute l'architecture.

4. Créer un certificat administrateur RA

Le serveur de la RA étant configuré pour exiger l'authentification client, il est nécessaire de fournir un certificat administrateur à présenter au serveur lors de la connexion HTTPS.

Ce certificat peut être généré de la même manière que le certificat du serveur publique.
C'est à dire en utilisant le scripte **openca-newcert** fournit par OpenCA.

Le certificat créé est ensuite stocké dans le répertoire `/usr/local/OpenCA/outbound` au format PEM.

Lors que l'administrateur se connectera au serveur HTTPS, le serveur apache demandera un certificat d'authentification. Le browser Netscape utilise uniquement le format de certificat

exportable PKCS#12, ce format permet d'inclure dans le même fichier le certificat et la clé privée associée, le tout protégé par un mot de passe d'exportation.

ftp://ftp.rsasecurity.com/pub/pkcs-12/pkcs-12v1.pdf

Le scripte suivant permet de convertir le certificat administrateur au format PKCS#12

Sh openca-browserexp

5. Importer le certificat administrateur dans le browser

Le certificat peut être ensuite importé dans le browser Netscape du poste administrateur.
Dans le menu `security->Certificate->Yours.`
Le bouton `import a certificate` permet d'importer le certificat ainsi que la clé privée dans le browser, le mot de passe d'exportation vous sera demandé à ce moment-là.

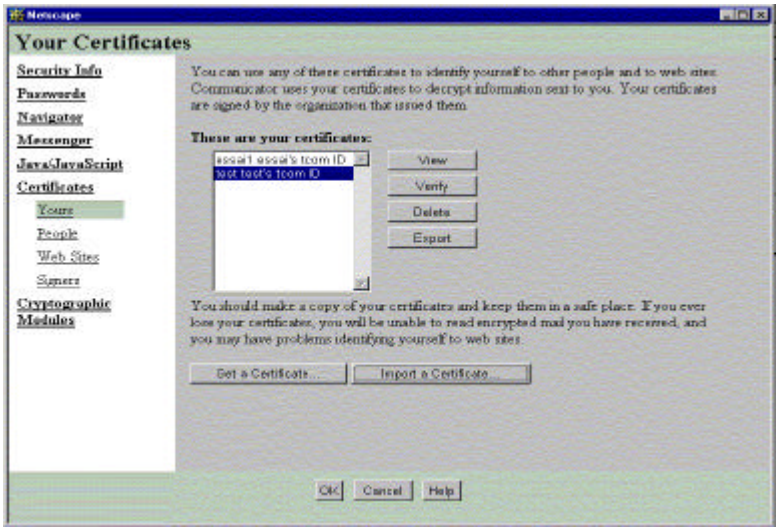

Netscape
Your Certificates
Security Info
Passwords
Navigator
Messenger
Java/JavaScript
Certificates
Yours
People
Web Sites
Signers
Cryptographic Modules
You can use any of these certificates to identify yourself to other people and to web sites. Communicator uses your certificates to decrypt information sent to you. Your certificates are signed by the organization that issued them.
These are your certificates:
View
Verify
Delete
Export
You should make a copy of your certificates and keep them in a safe place. If you ever lose your certificates, you will be unable to read encrypted mail you have received, and you may have problems identifying yourself to web sites.
Get a Certificate...
Import a Certificate...
OK
Cancel
Help

Figure 21 *Import du certificat administrateur*

Important : Il a été constaté que le script `openca-newcert` ne permettait pas de générer des certificats capables d'effectuer des signatures numériques au format PKCS#7. Cette signature est indispensable pour permettre un contrôle sur les requêtes traitée par les administrateurs de la RA.
Pour pallier à ce problème, le certificat administrateur réalisé ci-dessus doit être utilisé provisoirement jusqu'à la mise en œuvre totale de la PKI. Lorsque la PKI sera opérationnelle, un nouveau certificat administrateur devra être généré en suivant la procédure conventionnelle. Le certificat administrateur devra appartenir au groupe d'utilisateurs TcomCa Developper (ce groupe d'utilisateurs sera le seul à avoir accès à la RA).

6. **Modification du fichier de configuration du serveur de la RA**

Le certificat serveur au format PEM, ainsi que la clé publique au format PEM peuvent être copiés dans le répertoire **/etc/httpd/conf/ssl** du poste de la RA

Le fichier de configuration raSSL.conf doit ensuite être modifié de façon à utiliser le certificat créé à son intention pour toute connexion SSL. (Configuration 11)

Le fichier de configuration doit également indiquer le fichier root Ca à utiliser et la CRL de la CA, ces informations ont déjà été importée sur le poste de la RA. (point 2)
Le serveur de la RA doit également nécessité une authentification client.

```
#Certificat du serveur public
SSLCertificateFile conf/ssl/ra_server.pem
#Clé privée du serveur public
SSLCertificateKeyFile conf/ssl/ra_server_key.pem
#certificat de la CA
SSLCACertificateFile  /usr/local/RAServer/cacert.pem

#Emplacement de la CRL associée
SSLCARevocationFIle /usr/local/RAServer/crl/cacrl.pem
#L'authentification client est requise
SSLVerifyClient require
```

Configuration 11*Configuration avec certificat PKI (extrait raSSL.conf)*

A ce stade, il est possible de se connecter au serveur de la RA en présentant son certificat administrateur.

13.5.6 Droit d'accès au serveur RA

Le protocole SSL, utilisé pour s'authentifier au serveur de la RA, fournit des mécanismes puissants de chiffrage et d'authentification client, serveur. L'authentification client est requise pour s'assurer que seuls les utilisateurs identifiés par un certificat numérique puissent établir une connexion SSL avec le serveur.

A ce stade de la configuration, tous les clients pki possédant un certificat numérique pourraient sans autre se connecter au serveur de la RA et effectuer des modifications dramatiques sur les requêtes de certificats.

Pour éviter un tel cas de figure, des mesures de sécurité supplémentaires doivent été prises.

- L'administrateur doit posséder un certificat différent de ceux des clients, cette coloration particulière doit lui permettre d'être seul à pouvoir se connecter au serveur.
- L'administrateur doit ensuite utiliser un mot de passe et un «login » pour pouvoir accéder aux répertoires du serveur

Droit d'accès suivant le champ OU

La politique de la PKI permet de diviser les utilisateurs en trois catégories, `Tcom User`, `Tcom Developper`, et `Tcom employer`.(voire 13.4.1 Pré configuration OpenCa)
Ces catégories sont représentées par un attribut OU (Organization Unit) dans le DN du certificat. La coloration qui différenciera un utilisateur d'un administrateur est constitué du OU contenu dans le DN. Un administrateur doit faire partie du groupe `Tcom Developper`.

Le serveur apache met à disposition grand nombre de champ et d'expression booléen pour permettre une granularité dans les droit d'accès au serveur.
Ainsi il est tous à fait possible d'interdire l'accès aux possesseurs de certificat client dont le DN ne contient pas l'OU `tcom developper`.

```
<Location/>
SSLRequire(    %{SSL_CLIENT_S_DN_OU} eq « tcomCA Developper »)
</Location>
```

Configuration 12 *Limitation du droit d'accés par l'attribut OU (extrait raSSL.conf)*

Protection par mot de passe

La limitation d'accès par l'attribut OU a permis de réduire nettement les risques d'intrusion, mais cette méthode n'est pas suffisante car tous les possesseurs d'un certificat `tcom Developper` ne sont pas nécessairement des administrateurs RA. Il aurait été tout à fait possible de contrôler l'accès de façon plus fine en utilisant l'attribut CN (common name) de l'administrateur contenu dans son certificat comme filtre d'accès

Un choix autre a été fait pour différentes raisons :

- Si un nouvel administrateur RA entre en service, il sera nécessaire de modifier les droits d'accès dans le fichier de configuration du serveur, cette perspective n'est pas très souple.
- Les privilèges de l'administrateur sont entièrement contenus dans le certificat administrateur, cette politique ne correspond pas à la théorie PKI. En principe, un certificat ne devrait pas contenir de privilège.
- Toute la sécurité du système repose sur la confidentialité du certificat administrateur. Si pour une raison ou une autre un intrus arrive à s'emparer du certificat administrateur, (lunch time attack) tout le système peut être compromit.

Pour ces différentes raisons, l'administrateur doit s'authentifier d'une part à l'aide de son certificat numérique et deuxièmement par le bief d'un mot de passe administrateur

Apache fournit une méthode simple et efficace pour protéger les répertoires du serveur par un mot de passe, il s'agit des fichiers `.htaccess`

Ce fichier doit être créé et placé dans le répertoire à protéger, il protégera tous les sous-répertoire et les fichiers contenu dans le répertoire.

Il s'agit d'éditer un fichier nommé `.htaccess` qui a la structure suivante :

```
AuthUserFile /home/httpd/htdocs-raserver/.htpasswd
AuthGroupFile /dev/null
AuthType Basic
AuthName « Acces RA »
#contrôle d'accès sur le fichier .htacces et .htpasswd
<Files « .htaccess »>
order deny,allow
deny from all
</Files>
<Files « .htpasswd »>
order deny,allow
deny from all
</Files>
</Limit Get Post>
require user pgachet
</Limit>
```

Configuration 13 *droit d'accès au répertoires de la RA (extrait .htaccess)*

Ses paramètres sont les suivants :

`AuthUserFile` : Permet de stipuler au serveur où se trouve le fichier `username/password`, dans le cas présent, le fichier est également placé dans le répertoire à protéger.

`AuthName` : Message qui sera inclus dans la fenêtre d'accès au serveur

Les droits d'accès aux fichiers `.htaccess` et `.htpasswd` sont également protégés et limités.

Une fois ce fichier créé, il s'agit de définir un mot de passe correspondant à l'utilisateur.

La commande suivante permet d'ajouter le mot de passe correspondant à l'utilisateur dans le fichier `.htpasswd`.

```
Htpasswd –c .htpasswd {username}
```

```
Adding  password for username
New password :
Password
Re-type new password :
Password :
```

Le paramètre –c n'est à utiliser que la premier fois, il est ensuite possible d'ajouter des utilisateurs et leurs mot de passe sans ce paramètre.

Ce processus est indépendant des éventuels mot de passe administrateur pour des comptes ftp et autre, il n'a de l'effet que sur les fichiers à protéger !

Il est possible ensuite de se connecter à la RA en présentant son certificat d'administrateur, puis en entrant le login et le mot de passe de l'administrateur dans la fenêtre présentée. (figure 22)

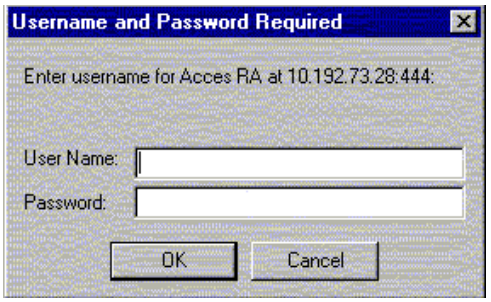


Figure 22 *Login administrateur*

Le mot de passe et le login sont transmis sur SSL, c'est à dire de manière totalement protégée.

13.5.7 Configuration serveur public

Il s'agit de configurer le serveur apache pour héberger le serveur public sur SSL.
Pour le serveur public de la PKI, le port TCP utilisé est le 443. Il s'agit du port well known de SSL. De cette manière, les clients pourront se connecter au serveur en utilisant HTTPS sans se soucier du port TCP utilisé.
Le serveur est configuré avec les contraintes suivantes :

- La configuration du mode SSL doit imposer l'authentification serveur, pour cela un certificat numérique doit être généré à l'intention du serveur publique.
- Le serveur public étant la partie publique et visible de la PKI, les clients ne doivent pas être contraints à présenter un certificat lors de la connexion SSL. D'autant plus que ce

serveur est justement voué à l'obtention des certificats, imposer l'authentification client à ce niveau deviendrait paradoxal. Le serveur public ne doit donc pas demander d'authentification client.

La configuration est semblable, mais nettement plus simple que la configuration du serveur de la RA.

La configuration du serveur public se limite à ces trois étapes :

1) Configurer le serveur apache sur le port 443 en utilisant en premier lieu les clés et le certificat serveur fourni par SSL.

2) Générer un certificat serveur à l'aide du script fourni par OpenCA

3) Modifier le fichier de configuration du serveur Apache afin d'utiliser les clés et le certificat créé à son intention.

1 . **Configuration du serveur publique**

Comme pour la configuration du serveur de la RA, il s'agit d'éditer un fichier de configuration appelé publicSSL.conf basé sur le fichier de configuration exemple de SSL `ssl.default-vhost.conf`. et de modifier les valeurs suivantes. (Configuration 14).

```
# Définition d'un virtual host sur le port 443
VirtualHost_default- :443

#Emplacement des pages html et des scripts CGI du serveur public
DocumentRoot /home/httpd/htdocs-public
ScriptAlias /cgi-bin/ /home/httpd/cgi-public
#Journal d'erreur du serveur, utilisé pour debuger
ErrorLog /var/log/http/public-error_log

#L'accès à ces emplacements doit être restreint
<Directory «/home/httpd/htdocs-public»>
  AllowOverride None
  Option None
  Order allow,deny
  Allow from all
</Directory>
<Directory «home/httpd/cgi-public »>
  AllowOverride None
  Option None
  Order allow,deny
  Allow from all
</Directory>
```

Configuration 14 *Virtual host serveur Public(extrait publicSSL.conf)*

Les paramètres spécifiques au serveur SSL suivant, doivent utiliser les certificats et les clés fournies par mod SSL (Configuration 15).

```
#Certificat SSL du serveur public
SSLCertificateFile conf/ssl/sslserver.pem
#Clé privée SSL du serveur public
SSLCertificateKeyFile conf/ssl/sslserver_key.pem
#certificat de l'autorité qui à signé le certificat SSL
SSLCACertificateFile  /etc/httpd/conf/ssl/cacert.pem
#L'authentification client ne doit pas encore être requise
SSLVerifyClient none
```

Configuration 15 *Paramètres SSL serveur Public (extrait publicSSL.conf)*

Il est alors possible de se connecter au serveur publique par l'URL suivant.

https:// 10.192.73.28

2. **Générer un certificat serveur**

Il s'agit ensuite de générer un certificat pour le serveur public. Cette opération est en tout point identique à celle effectuée dans la section «13.5.5 3.*Créer un certificat pour le serveur de la RA* »

3. **Modifier le fichier de configuration**

Une fois le certificat serveur créé et placé dans le fichier de configuration de SSL, le fichier de configuration publicSSL.conf peut être modifié, afin d'utiliser ce certificat. (Configuration 16)

```
#Certificat du serveur public
SSLCertificateFile conf/ssl/public_server.pem
#Clé privée du serveur public
SSLCertificateKeyFile conf/ssl/public_server_key.pem
#certificat de la CA
SSLCACertificateFile  /usr/local/RAServer/cacert.pem

#Emplacement de la dernière CRL
SSLCARevocationFile /usr/local/RAServer/crl/cacrl.pem
```

```
#Pas d'authentification client
SSLVerifyClient none
```

Configuration 16 *Paramètre serveur public (extrait publicSSL.conf)*

Étant donné que le serveur de la RA et le serveur public sont installé sur le même poste, l'emplacement du certificat root CA et de la CRL sont identique pour les deux serveurs.

Reconnaître le certificat comme CA de confiance dans le browser

Le certificat root CA est reconnu par le serveur apache comme signataire de confiance, mais cette confiance n'est pas encore gagnée par le navigateur Netscape.

Tous les clients PKI y compris les administrateurs devront faire reconnaître le certificat root CA par leur browser, cette procédure est indispensable, elle permettra de faire figurer la Pki dans la liste des signataires de confiance du browser. Bien évidemment, cette procédure doit être effectuée avec les contraintes de sécurité qui s'impose, il ne s'agit pas qu'une Pki pirate puisse se faire passer pour un organisme de confiance.

Cette procédure à été automatisé, pour permettre une installation simple du root CA dans le browser des clients.
Depuis l'interface publique de la CA https://10.192.73.28
Choisir le lien

Get CA certificate

Cette procédure va inclure le certificat root de la Pki dans la liste des différents signataire reconnus par le browser.

De cette manière, tcomCA devient un organisme de confiance, capable de vérifier des certificats numériques.
Cette procédure est complètement transparente à l'utilisateur, mais dans le fichier des outils de sécurité de netscape.(Figure 23)

```
Security-> signer
```

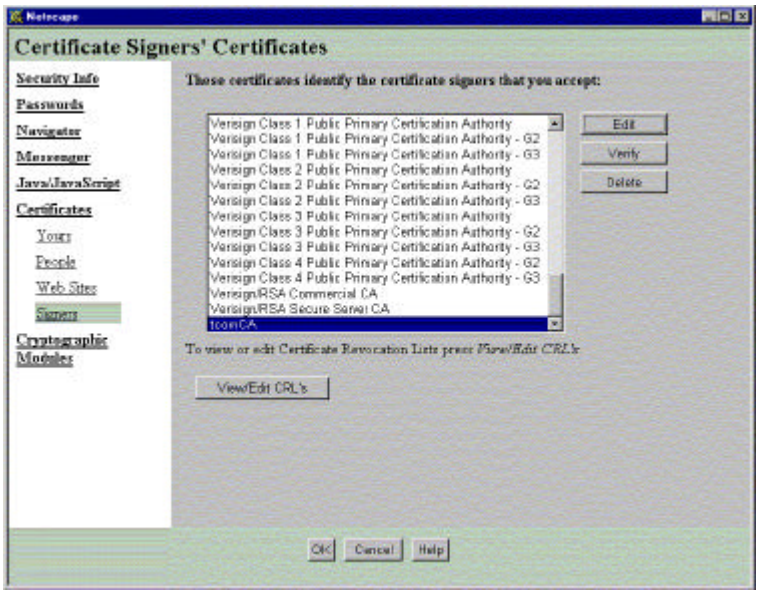

Certificate Signers' Certificates
Security Info
Passwords
Navigator
Messenger
Java/JavaScript
Certificates
Yours
People
Web Sites
Signers
Cryptographic Modules
Those certificates identify the certificate signers that you accept:
Verisign Class 1 Public Primary Certification Authority
Verisign Class 1 Public Primary Certification Authority - G2
Verisign Class 1 Public Primary Certification Authority - G3
Verisign Class 2 Public Primary Certification Authority
Verisign Class 2 Public Primary Certification Authority - G2
Verisign Class 2 Public Primary Certification Authority - G3
Verisign Class 3 Public Primary Certification Authority
Verisign Class 3 Public Primary Certification Authority - G2
Verisign Class 3 Public Primary Certification Authority - G3
Verisign Class 4 Public Primary Certification Authority - G2
Verisign Class 4 Public Primary Certification Authority - G3
Verisign/RSA Commercial CA
Verisign/RSA Secure Server CA
tcomCA
Edit
Verify
Delete
View/Edit CRL's
OK
Cancel
Help

Figure 23 ***New Authority***

On constate que la PKI tcomCA a été ajoutée à la liste des signataires de confiance. La signature de tcomCA sera désormais aussi crédible que toute autre autorité de certification mondialement reconnue par le browser

Cette procédure devra également être faite pas tous les clients. Pour pousser le degré de sécurité, il est possible et même souhaitable de transmettre par voie sûr, c'est-à-dire par courrier séparé, l'empreinte de la signature du root CA. De cette façon, le client vérifiera cette empreinte lors de la procédure de reconnaissance de l'autorité, évitant ainsi la reconnaissance d'une PKI pirate.

Cette sécurité peut paraître superflue, sachant que le serveur public est protégé par SSL, donc authentifié par certificat, mais est ce que tous les clients vérifient systématiquement les certificats des sites HTTPS ?

14 LDAP

14.1 configuration serveur LDAP

La Pki mise en œuvre doit être en mesure de générer des certificats numériques, mais également de les publier de façon efficace. Comme il été mentionné dans le tutorial (sécurité et PKI), cette publication utilise généralement le service d'annuaire fourni par LDAP.

OpenLdap est en principe déjà installé sur le poste de la RA.(13.2.4 Installation Openldap sur le poste de la RA).

Le fichier de configuration de ldap se trouve dans le répertoire
`/etc/openldap/slapd.conf`

L'administration de l'annuaire devra être faite depuis le serveur de la RA, pour cette raison l'annuaire doit être configuré de façon à être parfaitement intéropérable avec le serveur et les scriptes de la RA.

Le fichier de configuration de la RA se trouve à l'emplacement suivant.
`/home/httpd/cgi-raserver/conf/raserver.conf`

Dans la section consacrée à LDAP, il est nécessaire que certaines information coïncide avec les informations contenues dans le fichier `slapd.conf`

- Le nom du serveur (adresse IP) doit correspondre à l'annuaire LDAP utilisé, dans le cas présent le serveur LDAP est sur le même poste que la RA

- Le port utilisé est le port wellknow de LDAP 389, le serveur n'a volontairement pas été configuré sur SSL. Tous les utilisateurs doivent avoir accès aux informations contenues dans les certificats.

- Le paramètre «`basedn` », appelé «`suffix` » dans la terminologie ldap définit l'entrée de base de l'annuaire. Le formalisme traditionnel à été utilisé, soit l'union des attributs O=Organization et c=country .

- Le paramètre « `ldaproot` » appelé « `rootdn` » dans le fichier de configuration `slapd.conf` correspond au «dn » du manager ou de l'administrateur de la RA. Cette entrée doit bien évidemment être identique dans les deux fichiers de configurations.

- Le paramètre « `ldappwd` » ou « `rootpw` » correspond au mot de passe administrateur (manager), celui ci ne doit pas être passé en claire sur le réseau, la méthode pour réaliser ce chiffrage est expliqué par la suite.

Voici les extrait des deux fichier de configuration qui doivent correspondre.

```
## LDAP Section:
## =============
## LDAP Server Name
ldapserver 10.192.73.28:389
ldapport 389
## LDAP Maximum number of records returned by a query
ldaplimit 100
basedn "o=tcom, c=ch"
## DN du manager de l'annuaire
ldaproot "cn=Manager, o=tcom, c=ch"
## le mot de passe Manager n'est pas transmis en clair, le mot de
## passe ci dessous est passé par la fonction de hachage MD5
ldappwd "master_CA12"
## Let's define some Directory Env
## supposed to find there the bin/, sbin/ directory"
ldapbasedir "/usr"
```

Configuration 17 *section LDAP (extrait raserver.conf)*

Extrait du fichier de configuration de l'annuaire LDAP

```
#définition de l'entrée root de l'annuaire
suffix         "o=tcom,c=ch"
#définition de l'administrateur de l'annuaire
rootdn          "cn=Manager,o=tcom,c=ch"
# la variable rootpw contient l'empreinte MD5 du mot de passe
# Manager, cette procédure est explicitée par la suite
rootpw          {MD5}ijFYNcSNctBYg.
```

Configuration 18 *Configuration du serveur LDAP (extrait slapd.conf)*

Ces deux fichiers, comme tous les fichiers de configuration définis dans ce document, sont disponibles en annexe sur CD.

14.2 Hachage du mot de passe manager

Le manager devra se connecter à l'annuaire en transmettant un mot de passe sur le réseau. Pour éviter que le mot de passe soit transmis en clair, une empreinte du mot de passe est utilisée.
La commande suivante permet de calculer une empreinte MD5 d'un nom entré en ligne de commande.

Slappasswd –v –s –h{MD5}

L'utilisateur introduit la mot de passe manager et la commande ci dessus fournit l'empreinte MD5 résultante, cette empreinte est ensuite introduite dans le champ «`rootpw` » du fichier `slapd.conf`
De cette manière le mot de passe manager n'est jamais transmis en clair.

Lorsque le fichier de configuration de l'annuaire est modifié, il est nécessaire de redémarrer ldap.

`/etc/init.d/ldap restart`

14.3 Contrôle des droits d'accès

Il est important de limiter toute action qui pourrait entraîner une modification ou une suppression des données, seul l'administrateur doit être en mesure de modifier l'annuaire.

A cet effet il est possible de définir une liste de contrôle ACL (Access Contrôle List) a respecter lors des requêtes sur l'annuaire

Cette liste doit être définie dans le fichier de configuration `slap.conf` ou dans un fichier inclus.
Elle permet de contrôler quel type d'accès est défini sur quel objet, et par qui.

Les contrôles d'accès ont été définit de telle manière à ne laisser que le manager modifier ou introduire des données dans l'annuaire, mais la consultation de l'annuaire est possible pour tous.

Par exemple l'accès à l'attribut certificat, est possible pour tout les utilisateurs, l'accès est limité en lecture seule. Evidemment, les utilisateurs doivent être en mesure d'effectuer une recherche de certificats suivant un critère de recherche, comme le nom d'utilisateur par exemple. (Configuration 19)

```
access      to attr=usercertificate;binary
            by * read
            by * search
```

Configuration 19 *ACL (extrait slapd.conf)*

Lors de la liaison avec l'annuaire, une vérification va avoir lieu, le serveur ldap va rechercher la première ligne qui s 'applique à l'entrée en cours.

Lorsque la règle est trouvée, il est possible de déterminer quels sont les droits d'accès proprement dis, et là encore, c'est la première ligne qui s'applique, ce qui implique qu'il faut définir les règles les plus fines au début pour terminer par les plus grossières. Si aucune règle n'est définie, l'accès est celui par défaut

Suivant notre politique, l'accès par défaut est l'accès en lecture seule.

14.4 Initialisation de l'annuaire LDAP

Une fois l'annuaire correctement configuré pour interagir avec la RA, en respectant la politique d'accès défini, il est possible d'initialiser l'annuaire avec les premières données.
C'est-à-dire les entrées de l'organisation et les différents groupes d'utilisateurs, (tcom user, tcom developper, tcom employer). (Figure 24)

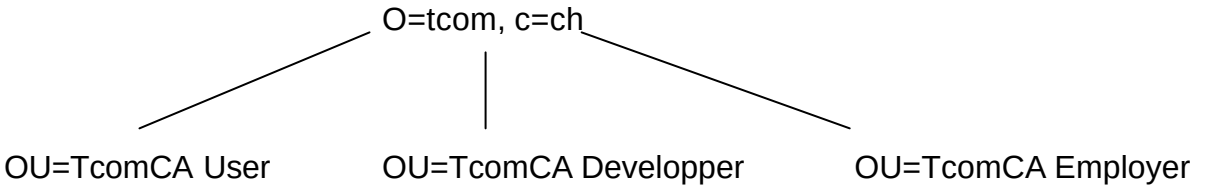


Figure 24 *Groupe d'utilisateur*

L'initialisation de l'annuaire peut être entreprise de deux manières.

- Composer un fichier LDIF contenant les classes d'objets à utiliser en suivant les contraintes du schéma, puis définir les différentes entrées à introduire.
  L'introduction du fichier dans l'annuaire est fait de façon manuelle par la commande
  `Ldapadd –x –D « cn=Manager,0=tcom,c=ch » -w –f {fichier.ldif}`

- La version 0.8.0 de openca fournit un script qui permet d'initialiser automatiquement de l'annuaire depuis le serveur de la RA.
  `RASERVER-> initLDAP`
  Cette fonction en plus d'initialiser l'annuaire avec les entrées voulues, introduit le certificat de la CA et la CRL courante dans l'annuaire. Cette méthode est bien plus simple à réaliser, l'inconvénient est que l'administrateur perd la vision d'ensemble de l'initialisation.

14.4.1 Initialisation par un fichier LDIF

Les fichiers LDIF (LDAP Data Interchange Format) permettent de représenter les données LDAP sous format texte standardisé, LDIF est basé sur un format texte ASCII. Ce fichier permet de faire des imports et des exports de données dans la base, par l'intérimaire de commande à l'intérieur du fichier.

Choix du schéma

Il est possible de spécifier dans le fichier ldif, en plus des entrées à intégrer dans la base, le schéma utilisé pour ces données, le schéma donne une définition précise des classes d'objets, de leurs attributs et de leurs syntaxe.

Le schéma définit également des contraintes sur les classes d'objet qu'il est nécessaire de respecter pour garantir l'intégrité de la base de données. Par exemple, une classe d'objet qui contient des attributs obligatoires doit permettre de s'assurer qu'aucun objet de cette classe ne puisse être sauvegardé dans l'annuaire sans valeur de cet attribut.

Le schéma ci-dessous (Configuration 20) définit la class `organizationalUnit`, qui requiert un attribut `OU`

```
objectclass ( 2.5.6.5 NAME 'organizationalUnit' SUP top STRUCTURAL
        MUST ou
        MAY   (   userPassword   $   searchGuide   $   seeAlso   $
businessCategory $
                x121Address       $       registeredAddress       $
destinationIndicator $
                preferredDeliveryMethod $ telexNumber $
                teletexTerminalIdentifier $
                telephoneNumber $ internationaliSDNNumber $
                facsimileTelephoneNumber $ street $ postOfficeBox $
                postalCode        $         postalAddress         $
physicalDeliveryOfficeName $
                st $ l $ description ) )
```

Configuration 20 *Exemple de schéma (extrait core.schema)*

Les différents schéma utilisés dans le cadre de la PKI sont définit dans les fichiers suivants

- Core.schema — Fichier principal OpenLdap(requis)
- Cosine.schema — schéma dérivé du standard x500
- Inteorgperson.schema — Attribut classique pour une organisation.

Les classes LDAP fonctionnent sur le principe d'héritage, c'est-à-dire qu'elles sont liées suivant une hiérarchie. Une classe peut avoir plusieurs classes filles, mais elle ne peut dériver que d'une seule classe.
Pour regrouper des classes il est parfois utile d'utiliser une classe abstraite. Les classes abstraites ne peuvent pas avoir d'instance mais les classes qui en dérivent peuvent en avoir. C'est notamment le cas de la classe abstraite `top` qui va regrouper toutes les autres classes..

Les classes suivante sont nécessaire à l'initialisation de l'annuaire

- `Top` Classe abstraite dont dérivent toutes les autres classes.
- `Organization` Classe définie pour être l'entrée de base de l'annuaire, cette classe requiert l'attribut o
- `OrganizationUnit` Classe définie pour diviser les utilisateurs suivant leurs fonctions dans la PKI, les unités suivantes ont été définies, `Tcom user`, `Tcom developper`, `Tcom employer`. Requiert l'attribut `ou`.
- CertificationAuthority Classe spécifique aux besoins de la PKI, elle requiert les attributs suivants :`authorityRevocationList`, `certificateRevocationList`, `caCertificate`.

Edition du fichier LDIF

L'étape suivante consiste à éditer un fichier LDIF permettant d'initialiser l'annuaire à l'aide des classes ci-dessus, tout en respectant les contraintes imposées par leurs schéma.

Concrètement, cela signifie que certains attributs doivent être défini alors qu'ils ne sont pas utilisé, c'est notamment le cas pour la class `certificationAutority` qui requiert l'attribut `authorityrevocationList` qui n'est pas utilisé, car la PKI mise en œuvre n'est pas en mesure de révoquer d'autre autorité de certification. Dans ce cas, la valeur de l'attribut restera nulle.(Configuration 21)

```
#entrée de l'organisation
Dn : o=tcom,c=ch
Objectclass : top
Objectclass : organization
Objectclass : certificationAutority
O : tcom
AuthorityrevocationList ;binary : » »
CertificateRevocationList ;binary : » »
CaCertificate ;binary : » »

#entrée de l'unité tcomCA user
dn : ou=tcomCA user,o=tcom,c=ch
```

```
objectclass : top
oranizatrionclass : organizationUnit
ou : tcomCa user
o : tcom
```

Configuration 21 *initialisation par fichier LDAP (extrait PKI.ldif)*

Ajout des entrées par ligne de commande

Pour intégrer les données contenues dans le fichier LDIF à l'annuaire, une commande ldap est prévue à cet effet

`Ldapadd –x –D « dn du manager » -w –f nom_fichier.ldif`

Paramètres :

`X` :simple authentification
`D` : définit la chaîne de connexion à l'annuaire
`W` :saisie du mot de passe manager
`F` : fichier ldif à insérer

Le manager devra introduire son mot de passe, celui-ci comme convenu sera passé par la fonction de hachage MD5, avant d'être transmis puis comparé avec le mot de passe manager défini dans le fichier de configuration `slapd.conf`

Si tout ç'est bien passé, l'annuaire LDAP est ainsi initialisé.

14.4.2 Initialisation automatique.

Il est possible avec la version 0.8.0 de OpenCA d'initialiser directement l'annuaire LDAP depuis le serveur de la RA, évitant ainsi les problèmes pratiques posés par l'initialisation par fichier LDIF

En réalité le scripte mis à disposition introduit les objets en utilisant des fonction openldap, comme `ldap_add` pour entrer les différentes informations dans l'annuaire.

Cette opportunité est très pratique pour autant que les spécifications des entrées soient les mêmes que celles définies par OpenCA.

Introduction des certificat utilisateur dans l'annuaire.

Par la suite, les certificats des utilisateurs, ainsi que la CRL sont introduits dans l'annuaire par l'intermédiaire de l'interface WEB de la RA. Ceci est possible car OpenCA met à disposition un script qui réalise cette action.

Le script se charge de l'utilisation de nouvelle classe comme `inetorgperson` définie dans le schéma `inetorgperson.schema.`

La classe ajoutée par le script est la suivante

- InetOrgPerson Cette classe est plus complète que sa super classe (`organizationalPerson`), elle permet de représenter une personne qui est lié à une association quelconque. Cette classe permet d'ajouter de nombreux attributs à la personne, comme son `mail,` son nom (`cn`), son prénom (`sn`) mais également l'attribut `userCertificate` qui contiendra le certificat numérique du client.

Une fois l'annuaire initialisé, les nouveaux certificats générés par la CA sont publiés dans l'annuaire LDAP depuis le serveur de la RA. L'administrateur de la RA doit introduire ces certificats par la commande `Add new Certs` depuis le site de la RA.

Dans la version actuelle de OpenCA, le contrôle de l'annuaire n'est pas encore implémenté. Mais les versions futures de OpenCA permettront un contrôle plus performant de l'annuaire depuis le serveur de la RA.

14.5 Client LDAP

L'intérêt de publier les certificats dans un annuaire LDAP réside dans la possibilité de pouvoir le consulter avec n'importe quelle client ldap, par exemple kldap, netscape etc.

Netscape étant un browser universellement distribué, c'est avec cet outils que la consultation de l'annuaire sera testée.

Dans le fichier de configuration `slapd.conf`, il est nécessaire d'ajouter le fichier de schéma suivant, pour que l'interaction avec netscape soit possible

```
Include /etc/openldap/schema/mull.schema
```

Il s'agit de configurer l'annuaire ldap de la pki dans le navigateur netscape.

- Ouvrir le carnet d'adresses via le menu communicator/Carnet d'adresse)
- Dans le menu du carnet d'adresses, choisir le menu Fichier/Nouveau répertoire (File/New Directory)
- Dans la fenêtre Propriété du serveur d'annuaires, enter les information suivantes

- Description : LDAP pki
- Serveur ldap : 10.192.73.28
- Rechercher la racine : o=tcom,c=ch

Puis ok pour fermer la fenêtre.

Étant donné que l'accès à l'annuaire est limité en lecture pour n'importe quel client, il est possible d'effectuer des recherches de client PKI suivant différents critères comme l'organisation, le nom etc.

Le résultat de la recherche fournira une fiche descriptive du client, avec des informations sur son certificat numérique. (Figure 25)

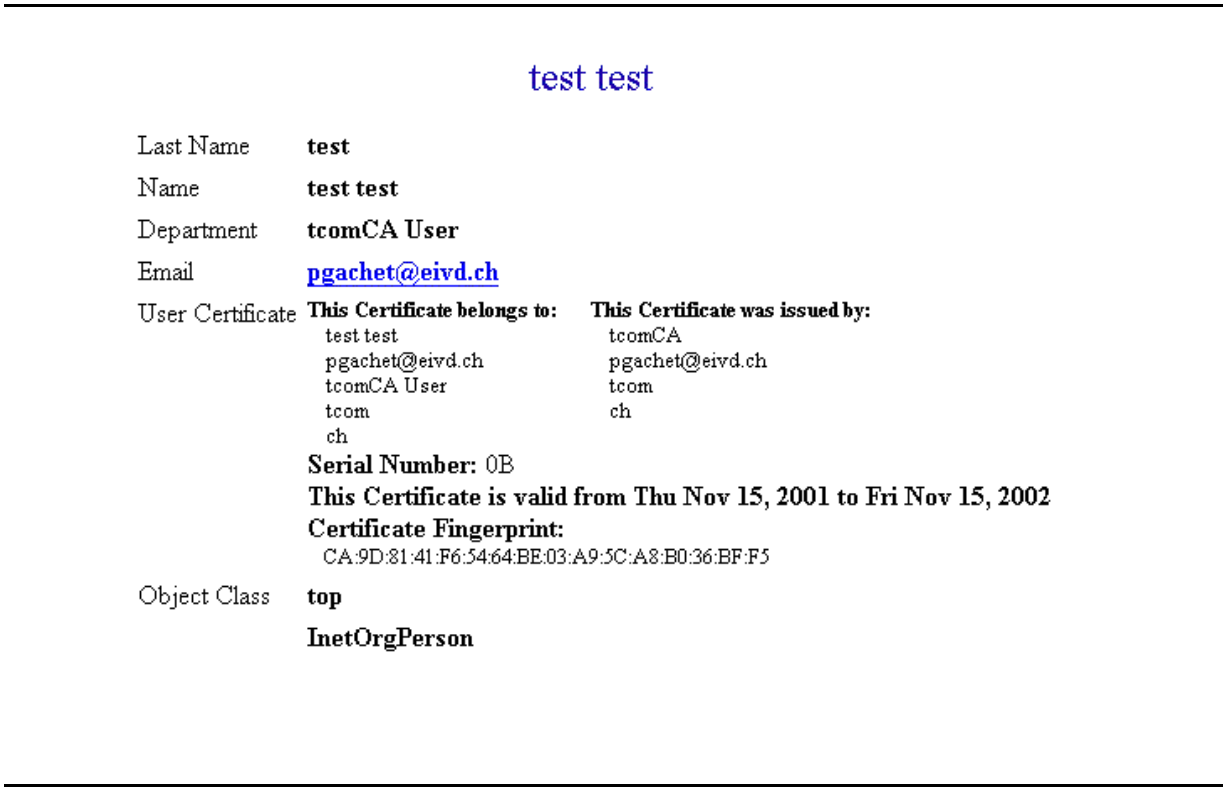
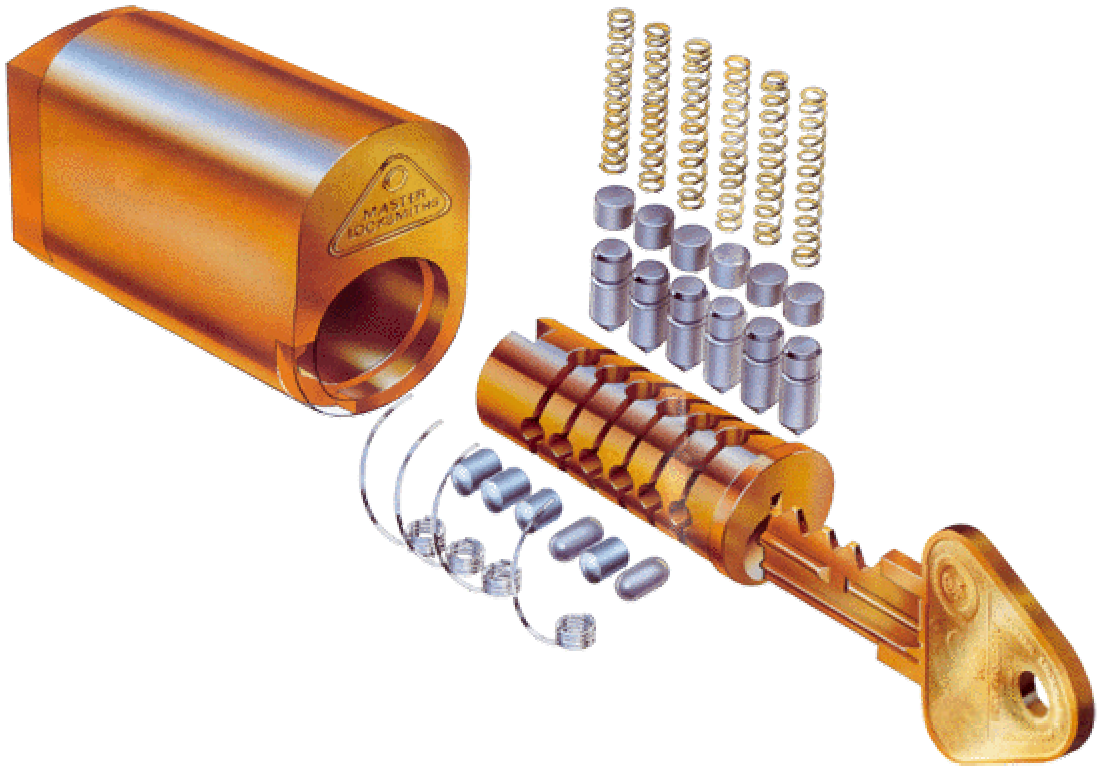


Figure 25 *client PKI sur LDAP*

La Pki est, à ce stade, parfaitement opérationnelle. Les différentes mécanismes permettant de gérer cette Pki sont détaillés dans le laboratoire PKI.

Laboratoire PKI



15 Laboratoire PKI

But : Ce laboratoire permet de s'initier à la manipulation d'un organisme PKI en effectuant les différentes étapes et procédures nécessaires à l'obtention d'un certificat numérique.

En deuxième temps, il sera entrepris des mesures et des tests sur l'utilisation des certificats numériques dans le cadre du protocole SSL.

Avant d'entamer ce laboratoire, il est nécessaire d'acquérir une connaissance théorique préalable sur les techniques de cryptographie et les PKI, à cet effet la lecture du tutorial « sécurité et PKI » et fortement conseillée.

15.1 Description de la maquette PKI

Une autorité de certification PKI est composée de différentes entités qui interviennent à tour de rôle dans la procédure de certification. (Figure26)

Ces entités sont physiquement représentées par des serveurs WEB. Ces serveurs contiennent différents scripts CGI qui, combinés à des paquetages de chiffrement, permettent de mettre en œuvre tous les mécanismes de chiffrage et de contrôle nécessaires à la gestion des certificats.

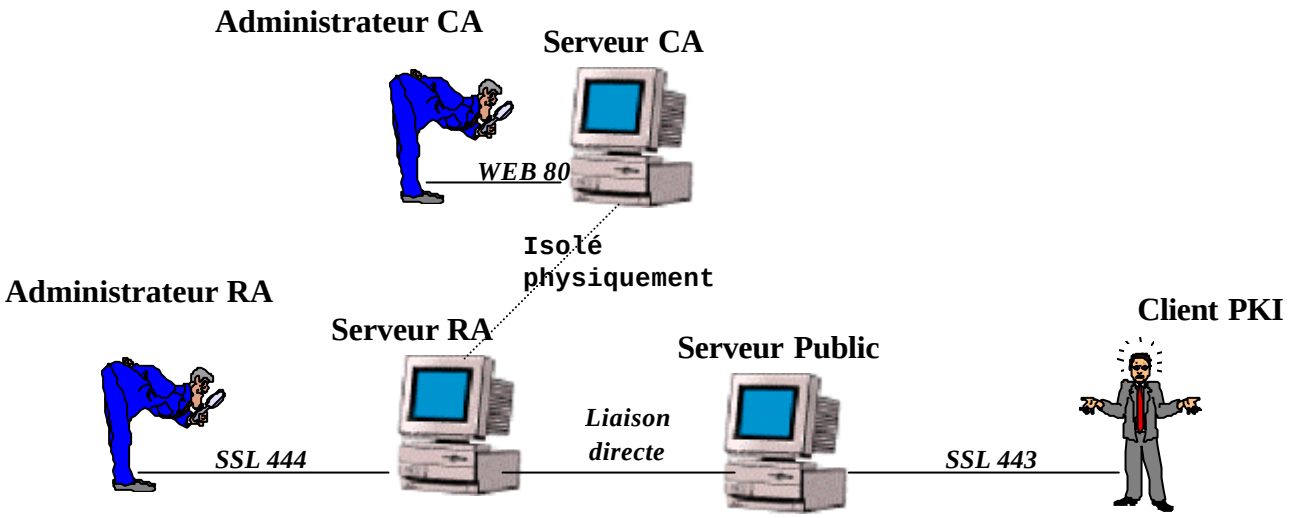


Figure 26 *Architecture OpenPKI*

15.1.1 Serveur Public

Cette entité est la partie visible de la PKI, elle permet aux clients d'entrer en contact avec l'organisme. Cette interface est utilisée par les clients pour transmettre des requêtes de certificats, mais également pour recevoir toutes les informations fournies par la PKI.
C'est-à-dire

- Réception du certificat numérique sollicité.
- Réception du certificat de l'autorité de certification
- Réception de la liste de révocation CRL
- Consultation des certificats numériques de tous les clients.

La connexion à ce serveur est protégée par le protocole SSL, ceci afin de garantir la confidentialité des données échangées entre les clients et le serveur publique.

15.1.2 Serveur RA

Ce serveur ne peut être accessible que par un administrateur authentifié, la connexion est ensuite protégée par SLL.
Le serveur RA permet de stocker, de contrôler et d'approuver les requêtes émisses par les clients. Ces requêtes seront ensuite transmises par voie sûre à l'administrateur de la CA, en vue d'une signature.

Ce serveur permet également de publier les certificats signé par la CA. La publication est faite de deux manières :

- Publication des certificats sur le serveur public.
- Publication des certificats dans un annuaire LDAP

Dans ce laboratoire, la publication des certificats par LDAP ne sera pas traitée. Les clients utiliseront donc uniquement l'interface publique.

Le serveur de la RA et le serveur public sont installés sur le même poste. Les échanges d'information entre ces deux entités se fait directement par fichier partagé.

15.1.3 Serveur CA

Ce serveur constitue la pièce maîtresse de l'organisme, c'est lui qui permettra de signer les requêtes de certificats approuvée par l'administrateur de la RA.
Pour protéger au maximum ce mécanisme, le serveur de la CA a été volontairement isolé du réseau.

L'accès à ce serveur doit impérativement être protégé physiquement. Seul l'administrateur de la CA doit être en mesure d'accéder au poste contenant le serveur. De ce fait, aucune sécurité

informatique supplémentaire n'est ajoutée. L'administrateur de la CA se connecte par une simple interface WEB sur le port 80, étant donné qu'il est le seul à pouvoir se connecter.

15.2 Rôles des membres PKI

15.2.1 Rôles des clients

Le rôle du client dans l'organisme, se limite à une tâche de sollicitation de service.
Pour effectuer une demande de certificat, celui-ci doit remplir un formulaire contenant différentes informations personnelles qui seront adjointes à son certificat.
Le client génère une paire de clés RSA, dont la partie publique sera transférée avec sa demande.

Le client choisit ensuite le groupe d'utilisateurs auquel il appartient, et également une zone d'enrôlement.
Le client attend ensuite que sa demande soit traitée, avant de la récupérer par l'intermédiaire du serveur public.

15.2.2 Rôle de l'administrateur de la RA

L'administrateur de la RA récupère les demandes fournies par les clients. Son rôle est de contrôler ces requêtes. Il a le droit de veto sur ces requêtes, c'est-à-dire qu'il peut rejeter toute requête qui ne correspond pas à la politique de certification de l'organisme.

Sans entrer dans le détails, le contrôle peut se limiter à la vérifier que le client possède bien une paire de clé RSA. Mais il peut tout aussi bien exiger que le client se présente physiquement avec une pièce d'identité valable.

Lorsque la demande de certificat a été approuvée par l'administrateur, celui-ci signe le document. La requête signée par l'administrateur compose un standard de requête au format PKCS#10 qui peut être transférée à la CA.

L'administrateur de la RA a également une tâche de publication, c'est lui qui est responsable de récupérer les certificats et les CRL signés, puis de les rendre accessibles aux clients.

15.2.3 Rôle de l'administrateur de la CA

Cette administrateur est hiérarchiquement le plus élevé de tout l'organisme. Sa responsabilité est plus grande que celle de l'administrateur de la RA car c'est lui qui signera le certificat proprement dit.

Sa tâche consiste à réceptionner les requêtes de certificat au format PKCS#10, puis à vérifier la signature de l'administrateur qui a émis cette requête.

L'administrateur de la CA ne contrôle en principe pas la validité des données contenues dans la requête. Il fait confiance au travail de contrôle effectué par l'administrateur de la RA.

Il validera ensuite la requête en la signant, le certificat ainsi formé sera conforme au standard de certificat X509. Le certificat doit être par la suite retourné à l'administrateur de la RA.

L'administrateur de la CA peut également révoquer des certificats qui ne sont plus conformes à la politique de certification de l'organisme. Son rôle consiste donc à générer périodiquement la liste des certificats révoqués CRL.

15.3 Zone d'enrôlement

Suivant le déploiement de la PKI, le nombre de requêtes des clients peut être relativement élevé, provoquant une surcharge de travail de la part de l'administrateur de la RA. Pour diviser le travail de contrôle de l'administrateur de la RA, l'administration de la RA est effectuée par plusieurs administrateurs.
A cet effet, trois zones d'enrôlement ont été définies, chaque zone étant gérée par un administrateur RA. Le client, lors de sa demande de certificat, choisit la zone d'enrôlement qui lui convient.
Cette division permet de répartir géographiquement les lieux d'enrôlement, tout en garantissant une bonne sécurité dans la procédure de contrôle.

15 .4 Manipulation de la PKI

15.4.1 Obtention du certificat CA root

La première étape dans l'obtention d'un certificat consiste à intégrer le certificat de l'autorité de certification dans le browser. L'autorité de certification doit être reconnue comme une autorité de confiance par le browser, celui-ci pourra dès lors faire une confiance totale dans les certificats émis par cet organisme, particulièrement celui que vous souhaitez obtenir. (Figure 27)

Cette procédure peut être mis en œuvre en activant le lien ***Get CA certificate***

Figure 27 *Réception du certificat Root*

Lors de cette procédure, le browser vous informe que cette procédure est critique du point de vue de la sécurité. (Figure 28)

New Certificate Authority - Netscape
New Certificate Authority
You are about to go through the process of accepting a Certificate Authority. This has serious implications on the security of future encryptions using Netscape. This assistant will help you decide whether or not you wish to accept this Certificate Authority.
Next>
Cancel

Figure 28 *Mise en garde du browser*

- Quelles sont les risques potentiels ?

Une fois que la CA est reconnue comme signataire de confiance, le client aura confiance dans tous les certificats signés par cet organisme. Cette procédure est la plus critique, car toute la politique PKI repose sur une chaîne de confiance ; si un intrus arrive à se faire passer pour

une autorité de confiance, il pourra duper le client avec des faux certificats. Et ainsi compromettre toute la sécurité mise en œuvre par SSL.

Lors de cette procédure, il est également nécessaire de définir les responsabilités de cette autorité de certification.

Dans notre cas, la CA doit être en mesure :

- De certifier des sites réseau
- De pouvoir certifier des E-mail, sous-entendu, pouvoir signer des documents.

Il est indispensable que la CA soit reconnue comme autorité capable de signer des documents, dans le cas contraire, le browser ne reconnaîtrait pas les certificats de cette CA.

15.4.2 Sollicitation d'un certificat

Pour obtenir un certificat numérique, il s'agit d'effectuer une demande auprès du serveur public de la CA.

Connectez vous à ce site

https://10.192.73.28

Sur la page principale, activez le lien suivant.

Request a Certificate

Le client peut choisir parmi trois possibilités de requêtes. (Figure 29)

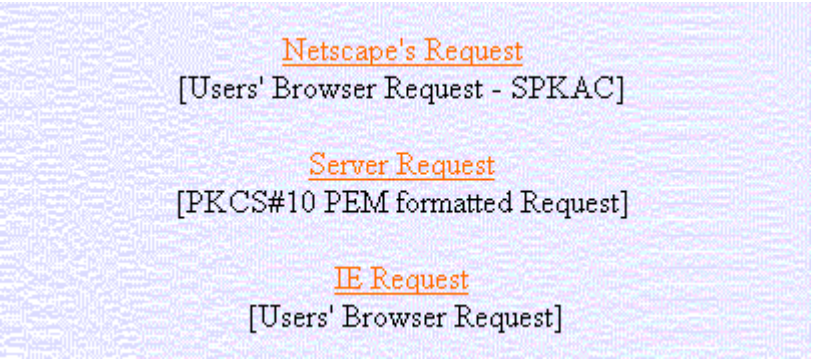


Figure 29 *Format de requêtes*

Les différents liens permettent de spécifier le type de requête désiré. Suivant le browser utilisé, la requête aura un format différent.

Utiliser le browser Netscape de préférence, car des incompatibilités ont été constatées avec IE.

Remplir le formulaire

Dans cette étape, le client doit fournir des informations personnelles qui seront adjointes à son certificat. On constate que le client peut choisir le groupe d'utilisateurs auquel il appartient.

Dans notre cas ***TcomCA User***

Le client peut également choisir l'autorité d'enregistrement qui va traiter sa requête.

Choisir ***System***

Une fois le formulaire correctement rempli, la requête est transmise à l'autorité d'enregistrement RA. Mais avant cela, la paire de clés RSA est générée.

Qui génère les clés ?
Les clés sont générées par le browser, suivant la version de celui-ci, la taille maximale des clés peut varier.
Où est stockée la clé privée ?
Les clés sont stockées à l'intérieur du browser, dans le fichier key3.db
Quelle clé est transmise avec le formulaire ?
Seule la clé publique est transmise

Comment améliore la confidentialité de la clé privée ?
Il est possible, et même conseillé, de chiffrer la clé privée à l'aide d'un mot de passe.
Security->Password->set password

15.4.3 Administration de la RA

Le formulaire ainsi crée est soumis à un administrateur, celui-ci a le droit de veto sur votre requête, c'est-à-dire qu'il peut refuser ou modifier la requête suivant la politique de certification.

Pour comprendre le rôle de cet administrateur, il est nécessaire de se connecter au site hébergent la RA en temps qu'administrateur.

Le site est protégé par une authentification SSL client, c'est-à-dire qu'il faut disposer d'un certificat numérique valide spécifique au groupe des administrateurs

Le certificat administrateur RA est disponible sur la disquette en annexe.

Raadmin.p12

Le format p12 ou PKCS#12 permet d'exporter le certificat dans le browser. En d'autres termes, ce format permet d'inclure dans le certificat la clé privée associée au certificat, le tout bien évidemment chiffré par un mot de passe.

Importer le certificat administrateur dans le browser

Pour importer le certificat administrateur dans le browser, les étapes sont les suivantes.

- Ouvrir les outils de sécurité communicator. Un raccourci à l'image d'un cadenas permet d'entrer directement dans le fichier d'information sur la sécurité. (Figure 30)

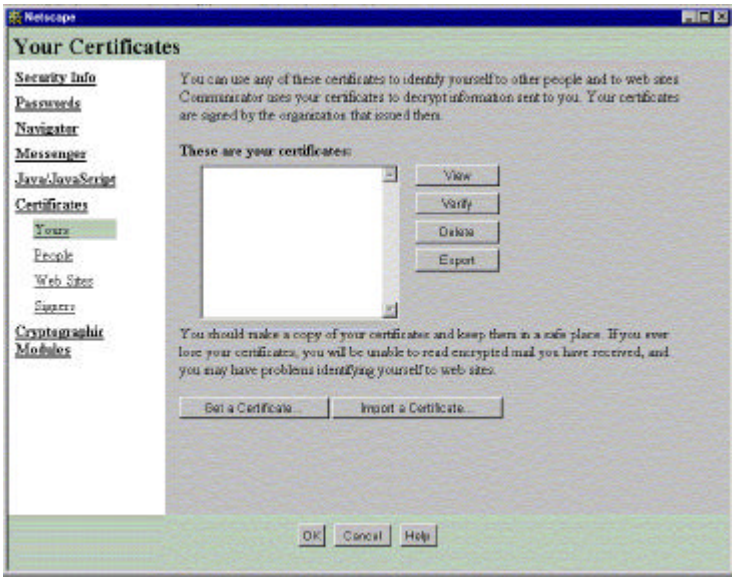

Netscape
Your Certificates
Security Info
Passwords
Navigator
Messenger
Java/JavaScript
Certificates
Yours
People
Web Sites
Signers
Cryptographic Modules
You can use any of these certificates to identify yourself to other people and to web sites. Communicator uses your certificates to decrypt information sent to you. Your certificates are signed by the organization that issued them.
These are your certificates:
View
Verify
Delete
Export
You should make a copy of your certificates and keep them in a safe place. If you ever lose your certificates, you will be unable to read encrypted mail you have received, and you may have problems identifying yourself to web sites.
Get a Certificate...
Import a Certificate...
OK
Cancel
Help

Figure 30 *Outils de sécurité*

- Puis cliquez sur le bouton ***Import a Certificate.***

Comme le certificat est au format p12, le browser demande le mot de passe pour importer le certificat.

Le mot de passe servant à l'import, export du certificats ce trouve sur la disquette..

Export.txt

- Une fois le certificat correctement installé et reconnu par le browser, il est possible de tester le certificat.

Toujours dans les outils de sécurité cliquer bouton ***verify***

Comment est réalisée cette vérification ?

Le browser vérifie en premier la date de validité du certificat, puis il déterminera qui a signé le certificat. Si le signataire est connu du browser, le certificat est jugé valide ; pour autant que l'autorité soit reconnue comme autorité capable de signer des certificats.

Connexion au serveur RA

Avant de tenter une connexion au serveur de la RA, il est nécessaire de connaître le « login » et le mot de passe administrateur. Ces informations vous seront demandées pour vous connecter.
L'information est disponible sur la disquette.

Login.txt

Vous disposez maintenant de tous les privilèges nécessaires pour tenter une connexion au serveur de la RA

Connectez-vous au serveur.

https://10.192.73.28 :444

Choisir un certificat d'authentification client, celui de l'administrateur.
Introduire le login et le mot de passe administrateur.

Vous êtes maintenant en mesure d'agir sur la partie enrôlement de la PKI

Dans le menu administration, choisir ***RAServer***.
Un nouveau menu vous permet de visualiser toutes les requêtes précédemment effectuées et également de contrôler les requêtes en cours.

Dans la rubrique ***pending request.***
Choisir ***certificate Request.***

La RA a été divisée en trois zones de responsabilité, pour permettre une gestion de la RA par plusieurs administrateurs.
Etant donné que votre requête a été transmise à la zone ***system***, c'est ici que doit se trouver votre requête.

Cliquer sur le numéro de série de la requête.

L'administrateur a une visibilité totale sur la requête, il sait par exemple quel est le type de requête utilisée, la taille de la clé publique et l'algorithme utilisé.

Il s'agit maintenant de transmettre la requête au format PKCS#10 à la CA.

Pour cela, la requête doit être signée par l'administrateur afin de pouvoir effectuer des contrôles sur la procédure. Cette signature doit impérativement être au format PKCS#7.

Avec quelle clé l'administrateur signe la requête ?

L'administrateur utilise la clé privée associée à son certificat pour signer la requête.

Avant de transmettre la requête à la CA, la requête approuvée peut être visualisée.

Dans le menu ***Request***,->***approved Request***.

Quel champ spécifie l'administrateur a signé le certificat ?
Un champ OP (ou Signer Certificate's Serial) définit le numéro de série du certificat administrateur qui a signé le certificat client.

Exporter le certificat client

Le média d'import export entre la RA et la CA est un moyen sûr, c'est-à-dire une disquette. Ce support permet de contrôler manuellement les échanges entre le serveur de la CA et le serveur de la RA.

Avant d'exporter la requête, il faut s'assurer que le poste physique où est situé le serveur de la RA contienne une disquette dans le lecteur.

Puis sur ce poste, la commande suivante permet de monter la disquette.

Mount /dev/fd0 /mnt/floppy/

Ensuite, depuis l'interface WEB de la RA.

Choisir ***Raserver Admin.***

Puis, dans le menu ***Outbound***

Choisir ***Export Request***

La disquette doit être manuellement extraite du lecteur, sans oublier de démonter le disque.

Umount /mnt/floppy

Pourquoi est il important que la CA soit physiquement isolée du réseau ?

Etant donné que le poste de la Ca contient les méthodes et les outils cryptographiques qui permettront de signer tous les certificats, il est indispensable que ce poste soit protégé contre toute tentative d'intrusion. La façon la plus sûr de le protéger et de l'isoler complètement du réseau.

15.4.4 Administration de la CA

L'interface de la CA est également accessible depuis un serveur WEB. Etant donné que le serveur est isolé du réseau, il est nécessaire de se connecter directement depuis le poste de la CA.

http://localhost

Pourquoi ce serveur n'est pas protégé par SSL, alors qu'il héberge l'entité la plus critique de l'organisme PKI ?

Etant donné que ce poste est isolé physiquement du réseau, la protection physique et nettement plus efficace que toute autre protection logiciel supplémentaire.

Introduire la disquette, puis monter le lecteur

Mount /dev/fd0 /media/floppy –o uid=wwwrun

La commande est différente, car la distribution de Linux est une SUSE.

Dans le menu ***Input and Output***

Choisir ***Import Request***

La requête peut ensuite être traitée depuis le menu ***Pending Request***

Choisir ***Certificate Request***

On constate que l'administrateur de la CA ne peut pas modifier les champs de la requête, il peut tout au plus modifier le profil appliqué au certificat.

Quel est le format de la requête ?
Requête au format PKCS#10

Lorsque la requête sera approuvée, l'administrateur de la CA devra la signer pour former un certificat numérique authentique.

Le mot de passe pour signer le document se trouve sur la disquette.

Capass.txt

A ce stade, le certificat a passé toutes les étapes de contrôle, il est considéré comme valide. L'étape suivante consiste à transférer le certificat à la RA, en vue d'une publication.

Dans le menu ***Input and Output***
Choisir ***Export Certs***

Cette commande a pour but de transférer le certificat sur la disquette. Celle-ci peut être réintroduite dans le poste de la RA.

L'administrateur de la RA peut récupérer les certificats depuis le menu ***Inbound***
Choisir ***Import New Certs***

Le certificat est maintenu provisoirement dans une base de données de la RA, jusqu'à ce que le client le récupère.

15.4.5 Réception du certificat

Le client ayant émis une demande de certificat doit être informé que son certificat est prêt, ceci peut être entrepris de deux manières.

- Le client peut consulter la liste des certificats valides sur le serveur public.
  Valid certificates liste

- L'administrateur de la Ra informe le client par e-mail.

Dans les deux cas, le client doit connaître le numéro de série de son certificat. C'est ce numéro qui devra être introduit dans la boite de dialogue en temps voulu.

Depuis l'interface publique, choisir

Get Requested Certificate

Puis introduire le numéro de série de votre certificat.

Le certificat est automatiquement introduit dans le browser, sans information supplémentaire.

On peut vérifier la procédure depuis la rubrique sécurité de netscape.

Certificates-> Yours.

Le certificat peut être importé et exporté au format PKCS#12 de la même manière que le certificat administrateur fourni sur la disquette.

- Quel mécanisme permet d'assurer que seul le client ayant émis la requête de certificat peut s'approprier le certificat ?

  Seul le client qui à la clé privée associée au certificat est en mesure d'importer le certificat dans le browser. Il est donc nécessaire de récupérer le certificat depuis le même poste, avec le même browser que celui utilisé pour effectuer la demande.

- Essayer d'introduire le numéro de série d'un autre certificat.

15.4.6 Liste de révocation

Un certificat numérique, comme toute pièce d'identité, comporte une fourchette de validité. Lorsque il sera présenté à une application, celle-ci vérifiera la date de validité du certificat .

Mais à la différence d'une carte d'identité, le certificat peut être révoqué. Cette procédure de révocation peut être entreprise pour plusieurs raisons.

- Les informations personnelles du client ont changé, le certificat n'est donc plus à jour
- Le client a perdu la clé privée associée au certificat
- L'autorité de certification a jugé que le client ne méritait plus d'être certifié, pour différentes raisons.

Dans tous les cas, le certificat ne peut pas être retiré du réseau, car des copies multiples existent à différents endroits. Pour cette raison, l'administrateur de la CA est responsable de publier régulièrement une liste qui contient tous les certificats révoqués.

Les applications qui reçoivent un certificat doivent consulter cette liste pour s'assurer que le certificat présenté n'a pas été révoqué.

Révoquer un certificat depuis la CA

Cette opération ne peut être faite que par l'administrateur de la CA.

Depuis le poste de la CA

Dans le menu ***Certificates***

Choisir ***Valid Certificates***

Choisir un certificat dans la liste, et révoquez le.
Attention, cette procédure est lourde en conséquence, révoquez uniquement un certificat que vous avez généré ! ! !

Le mot de passe administrateur vous est demandé pour révoquer un certificat.

Créer la liste de révocation CRL

Les certificats révoqués sont contenus dans la base de données de la Ca. Elle peut être consultée.

Dans le menu ***Certificates***
Choisir ***Revoked Certificates***

Expliquer précisément la différence entre un certificat échu et un certificat révoqué ?

Le certificat échu à atteint sa date de validité, ce qui signifie qu'il n'est plus valable, cette date de validité est contenu dans le certificat. Un certificat révoqué n'a pas forcement atteint sa fin de validité, mais il a été jugé non valide pas l'autorité de certification. Dans les deux cas, les applications doivent rejeter le certificat, dans un cas la date est contenue dans le certificat ce qui facilite le contrôle, mais dans le cas de la révocation, rien dans le certificat ne permet de discerner un certificat révoqué d'un certificat valide.

L'étape suivante consiste à créer une liste contenant tous les certificats révoqués depuis la mise en œuvre de la CA.

Dans le menu `CRL`

Choisir ***Issue new CRL***

La liste de révocation a une structure semblable à tout certificat numérique, elle comporte en plus de l'identité de l'administrateur de la CA, la liste de tous les certificats révoqués.
Comme pour le certificat numérique, la CRL doit être signée par l'administrateur. La CRL a une durée de validité limitée.

L'administrateur est donc contraint de publier régulièrement des CRLs, pour permettre aux applications de contrôler à tout moment la validité des certificats présentés.

La CRL générée doit ensuite être transmise à la RA de la même manière que tout autre certificat, c'est-à-dire de façon sûre, par disquette.

Dans le menu `Input and Output`

Choisir ***Export CRL***

Publication de la CRL

L'administrateur de la RA doit être en mesure de publier cette CRL, et de la distribuer aux client PKI.

Depuis l'interface de la RA, menu ***Inbound***

Choisir ***Import CRL.***

La CRL est automatiquement transmise au serveur publique , elle est prête à être téléchargée par les clients et les applications compatibles PKI.

Chargement de la CRL pour les clients

Il existe plusieurs façons de charger la CRL dans le browser.

1) Depuis l'interface public de la pki
2) Directement depuis Netscape

1. Depuis l'interface publique

Sur la page html du serveur public, choisir le lien.

Certificate Revocation Lists

Puis choisir le type de format de liste

.

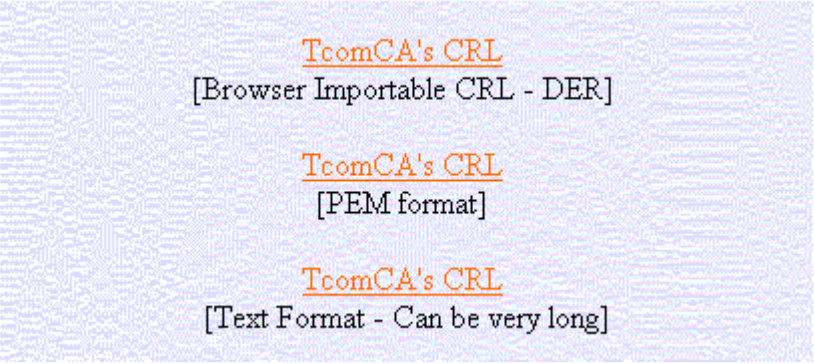


Figure 31 *Format des CRL*

Le format DER permet d'importer automatiquement la CRL dans le browser, alors que les types PEM et Text peuvent uniquement être consultés à titre informatif.

2. Depuis le browser

La CRL peut être chargée directement depuis le browser Netscape (Figure32)

Dans les propriétés sur la sécurité de netscape, dans le menu ***Certificate->signers***

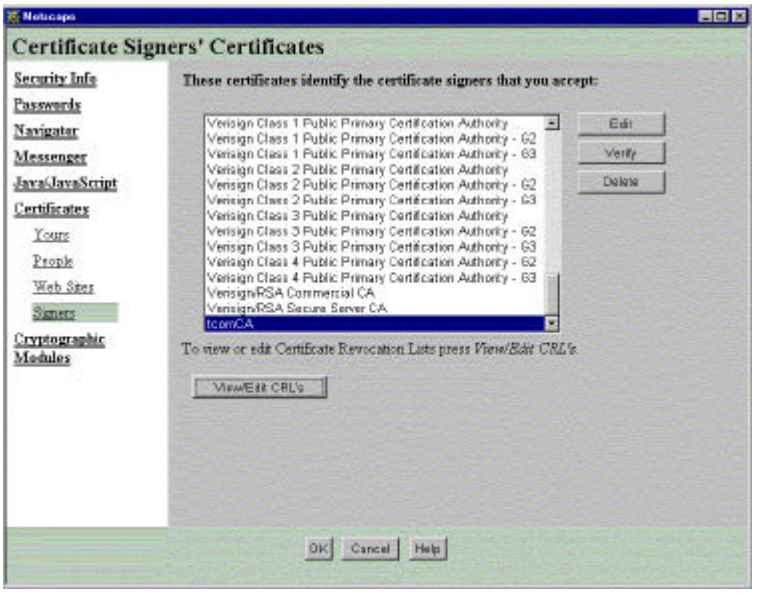

Certificate Signers' Certificates
Security Info
Passwords
Navigator
Messenger
Java/JavaScript
Certificates
Yours
People
Web Sites
Signers
Cryptographic Modules
These certificates identify the certificate signers that you accept:
Verisign Class 1 Public Primary Certification Authority
Verisign Class 1 Public Primary Certification Authority - G2
Verisign Class 1 Public Primary Certification Authority - G3
Verisign Class 2 Public Primary Certification Authority
Verisign Class 2 Public Primary Certification Authority - G2
Verisign Class 2 Public Primary Certification Authority - G3
Verisign Class 3 Public Primary Certification Authority
Verisign Class 3 Public Primary Certification Authority - G2
Verisign Class 3 Public Primary Certification Authority - G3
Verisign Class 4 Public Primary Certification Authority - G2
Verisign Class 4 Public Primary Certification Authority - G3
Verisign/RSA Commercial CA
Verisign/RSA Secure Server CA
tcomCA
Edit
Verify
Delete
To view or edit Certificate Revocation Lists press View/Edit CRL's
View/Edit CRL's
OK
Cancel
Help

Figure 32 *Netscape CRL*

Choisir l'autorité de certification de tcom, ici tcomCA.
A ce stade, il est possible de voir ou de charger la CRL associée à l'autorité de certification sélectionnée.

Bouton ***View/Edit CRL's***

Si une liste de révocation est déjà chargée dans le browser, la liste du browser contiendra cette CRL.(Figure 33)

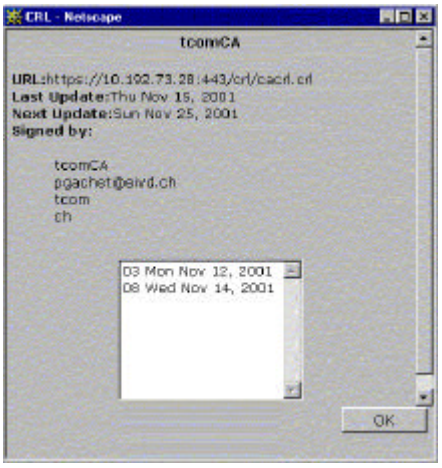


Figure 33 *tcomCA CRL*

Il est possible de visualiser cette liste ou de la recharger.

Comme prévu, cette liste contient le numéro de série de tous les certificats révoqués par l'administrateur de la CA.

Si aucune liste n'a été chargée, il est sans autre possible d'indiquer au browser l'url de cette liste.
Cette liste est accessible depuis le serveur publique de la PKI.

https://10.192.73.28:443/crl/cacrl.crl

En principe, tous les certificats émis par l'autorité de certification TcomCa contiennent cette URL dans un des champs extension. Malheureusement, Netscape dans sa version 4.75 n'extrait pas automatiquement cette information.

Que se passe t il si le client n'a pas la version à jour de la CRL ?

La CRL comme tout autre certificat numérique, contient une date de validité, si cette date est atteinte, le browser peut refuser une connexion, car il y a un risque potentiel que le certificat présenté aie pu être révoqué.

Qui effectue le contrôle de la CRL pour une connexion HTTPS ?

La vérification peut être effectuée d'une part par le serveur HTTPS qui contient également la liste CRL, mais le browser effectue également un contrôle de la CRL dans le cadre d'une connexion HTTPS.

15.5 Etudes d'échange de certificat dans le cadre du protocole SSL

Le but de cette étape consiste à tester l'utilisation des certificats numérique obtenu par la PKI dans le cadre concret de SSL.

15.5.1 Etude du protocole SSL

SSL /Secure Sockets Layer) est le standard actuel pour toutes les transactions sécurisées à travers Internet, les premières versions de SSL ont été développées par Netscape. Actuellement, l'IETF a spécifié un standard de protocole sécurisé se basant sur SSLv3, ce standard est appelé TLS(Transport Layer Secure).

Protocole SSL

Le protocole SSL utilise toutes fonctionnalités offertes par TCP/IP pour apporter une sécurité aux couches application. (Figure 34)

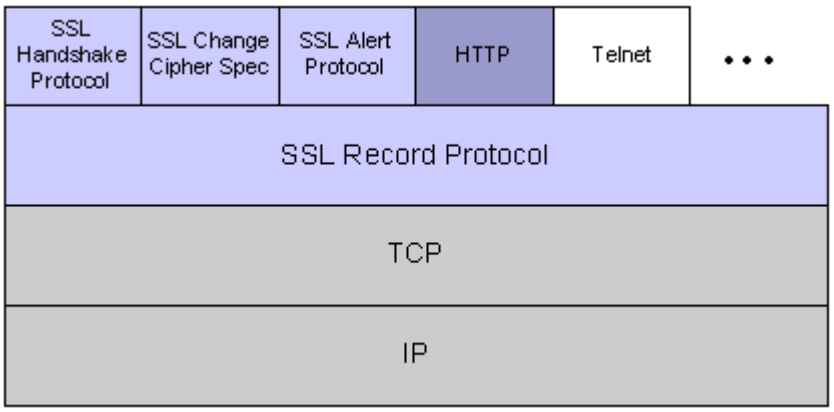


Figure 34*Couche SSL*

Une connexion sécurisée par SSL doit permettre d'assurer les contraintes suivantes

- Authentification du serveur.
- Authentification du client.
- Chiffrement des données
- Intégrité des données

Principe de base de SSL

Pour aboutir à une communication sécurisée, les deux entités en présence doivent s'authentifier. Cette authentification se base sur un chiffrage RSA et un échange de certificats x509.

Les certificats contiennent les clés publiques des correspondants. Le client extraira la clé publique de son interlocuteur et transmettra une clé symétrique chiffrée avec la clé publique de son correspondant.

La communication pourra par la suite être entièrement chiffrée par cette clé symétrique connue de part et d'autre.

15.5.2 Connexion SSL sans authentification client.

En premier lieu la manipulation consistera à établir une connexion à un serveur HTTPS qui n'exige pas d'authentification client. Le serveur public de la PKI correspond à ce profil.

Etablir une connexion HTTPS avec le serveur sécurisé à l'adresse suivante.

HTTPS://10.192.73.28

Effectuer une mesure du trafic sur ce poste.

- Quel est le port well known utilisé ?
  443

- Etudier les différentes étapes du protocole : Représenter les échanges sur un diagramme en flèche.
  http://sw00d.free.fr/crypto/pages/ssl.htm

- Définir tous les champs des messages client hello et client serveur.

- Décrire de manière schématique comment le client va vérifier le certificat présenté par le serveur.

- Il manque une information importante pour aboutir à un contrôle sans équivoque du certificat, qu'elle est-elle ?

L'autorité de certification n'est pas reconnue par le browser, la signature de la CA qui a émit le certificat serveur ne figure pas dans la liste des signatures de confiance du browser.
Outils->information sur la sécurité->signataire

Pour que la signature figure dans cette liste il est nécessaire d'effectuer une procédure de reconnaissance d'autorité de certification.

15.5.3 Connexion SSL avec authentification client

Le serveur de la RA procède à une authentification client avant d'établir une connexion, pour étudier et mettre en évidence les différents mécanismes de contrôle, connectez-vous en utilisant les différents certificats disponible sur la disquette.

Raadmin1.p12

- Pourquoi ne peut on pas utiliser ce certificat pour ce connecter, comparer le contenu de ce certificat avec le certificat Ra admin.p12.

  Seul les certificats délivrés pour Tcom Developper permettent d'accéder au serveur.

- Quelle est la granularité du contrôle des droits d'accès ?
  La granularité porte sur le champs OU (Organization Unit) du DN

- Cette limitation pour gérer le droit d'accès est elle suffisante ?
  Non, car tous les possesseurs d'un certificat Tcom Developper ne sont pas nécessairement des administrateurs. Un mot de passe est une façon simple et efficace de limiter l'accès aux répertoires de la RA

Connectez vous en utilisant le certificat **Raadmin2.p12**

- Pourquoi n'est-il pas possible de se connecter , est-ce un problème de droit d'accès ?
  Il ne s'agit pas d'un problème de droit d'accès, mais d'une erreur lors de la connexion. Le serveur a refusé l'identité du client et a fermé la connexion

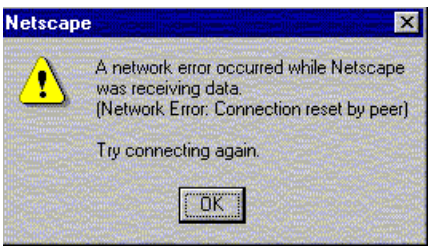

Netscape
A network error occurred while Netscape was receiving data.
(Network Error: Connection reset by peer)
Try connecting again.
OK

Figure 35 *Erreur de connexion SSL*

- Etudier le contenu du certificat, principalement le numéro de série du certificat et expliquer la cause de l'erreur signalée par Netscape.
  Le numéro de série du certificat client figure dans la liste de révocation CRL, ce certificat n'est plus valide, donc le serveur refuse catégoriquement la connexion, ce qui provoque une erreur dans le navigateur.

[1] SSL : Secure Sockets Layer
http://www.guill.net/reseaux/Ssl.html
[2] Mod_SSL
Http://modssl.org/docs/2.8/ssl_overview.html

16 Intégration de la technologie PKI dans le cadre d'une solution VPN basée sur Ipsec

16.1 Introduction

La mise en œuvre d'une PKI à permis de fournir un moyen sûr de distribution de certificat numérique, de façon centralisée. Ces certificats sont utilisés d'une part comme moyen d'authentification, par exemple lors d'une connexion HTTPS, mais leurs potentialités sont nettement plus vastes. Ils rendent possible la signature de mail, le chiffrage de mail, etc..

Le principal intérêt qui nous a poussé à mettre sur pied un tel organisme est l'interaction possible de la PKI avec un VPN et plus particulièrement l'utilisation des certificats numériques comme moyen d'authentification pour établir une connexion sur le VPN.

Choix d'un protocole

Le choix d'un protocole permettant de déployer une solution VPN est basée sur le travail de diplôme effectué par C.Tettamanti (Banc de test VPN 2000)

Il existe différents protocoles permettant la mise en œuvre d'un VPN, par exemple PPTP, SSL, L2TP, Ipsec. Malgré son extrême lourdeur, seul le protocole Ipsec a été retenu pour mettre en œuvre une solution réaliste. Ce choix a été fait pour plusieurs raisons

- Ipsec fournit des mécanismes de chiffrage puissants déployés au niveau réseau. Il permet d'encapsuler les flux de données de toutes les applications utilisant le protocole IP, il est donc parfaitement efficace pour les applications Internet. Cette transparence n'est pas aussi évidente avec des protocoles comme SSL ou SSH qui travaille au niveau application.

- Ce protocole a atteint un grand niveau de maturité. De nombreux fournisseurs l'on parfaitement intégré dans leurs produits. Il peut donc rendre intéropérable des systèmes VPN hétérogènes.

- Il propose plusieurs moyens d'authentification dans sa phase d'identification, entre autre l'authentification forte par certificat numérique. Ce moyen d'authentification n'est pas possible avec un protocole comme PPTP.

Choix d'une implémentation libre d'Ipsec

Le cahier des charges prônait une solution utilisant des logiciels libres tournant dans un environnement ouvert comme LINUX.
Le choix d'une implémentation libre d'Ipsec est basée sur les tests effectués par C.Tettamanti. sur le produit Freeswan.

Bien que ce produit ne fournisse pas une implémentation complète d'Ipsec, l'authentification par certificat n'étant pas réalisable, il à été retenu malgré tout, car son développement est en pleine croissance. De nouvelles versions, toujours améliorées, poussent à croire que ses lacunes seront comblées dans un avenir proche.

Ce choix a été particulièrement motivé par le travail effectué en parallèle par un groupe de l'université en sciences appliqué de Winterthur. Leur travail s'est justement porté sur la possibilité d'utiliser des certificats numériques dans le cadre de Freeswan.
http://strongsec.com

Ce groupe fournit un patch pour freeswan qui permet une reconnaissance du format x509 dans l'implémentation du protocole IKE.
Le paquetage contient également différent outils permettant d'extraire des informations des certificats numériques.

16.2 Analyse d'échange des clés et d'authentification pour Ipsec

Les mécanismes d'authentification et d'échange des clés dans le protocole Ipsec ont été étudiés de façon théorique lors du travail de semestre. Le résultat de l'étude est contenue dans le dossier « Gestion des clés pour Ipsec ».

Suite à cette étude, ces mécanismes ont pu être mis en pratique et analysés à l'aide de différentes configurations de freeswan.

Cette analyse porte uniquement sur la partie authentification du protocole IPsec, car les autres étapes conduisant à une connexion Ipsec ont déjà été étudiées et publiées par C.Tettamanti.

Ipsec utilise le protocole IKE pour établir des associations de sécurité de façon automatique, c'est durant cette phase que l'authentification et l'échange des clés à lieu.

Un mode manuel d'échange de clés est possible, mais son utilisation est fortement dépréciée par les développeurs de freeswan. de ce fait seule IKE automatique à été utilisé.

Le protocole IKE définit différentes alternatives d'authentification tirées du cadre générique ISAKMP. Ces différentes méthodes sont possibles car ISAKMP ne définit pas de déroulement fixe, il fournit un certain nombre de blocs, nommés payload ISAKMP. Notamment un bloc `Identification` qui contient les données qui serviront à

l'indentification des tiers, ou mieux un bloc `certificate` qui permet de transporter toutes sortes de certificats numériques.

16.2.1 Echange avec protection de l'identité (identity Protection)

A partir des blocs utilisés, ISAKMP définit des types d'échanges, dans le cas du VPN seul le type d'échange `Identity Protection Exchange` a été utilisé. Ce type d'échange est le seul à procéder à un échange de clés avant l'authentification, protégeant donc l'identité des interlocuteurs, il s'agissait du type d'échange par défaut de freeswan.
L'échange est composé de la séquence de messages suivants, dans une configuration Host to LAN.

SA	Security Association Payload
KE	Key Exchange Payload
ID	Identity Payload
Nonce	Nonce Payload
Auth	Authentification (SIG or HASH)

Figure 36 *Notation pour les échanges ISAKMP*

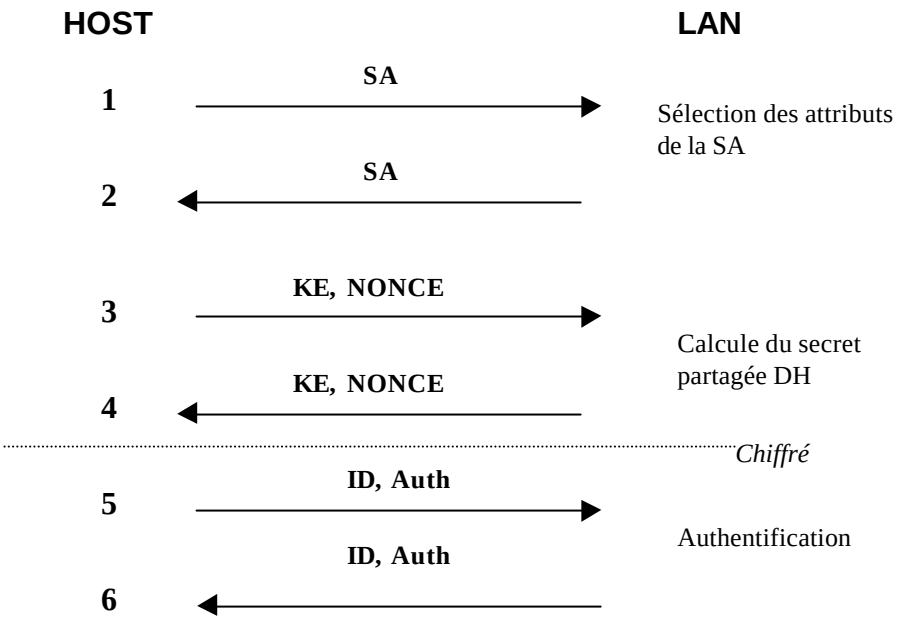


Figure 37 *Echange identity protection*

La méthode d'authentification est choisie durant la phase 1 du protocole, c'est-à-dire lors des deux premiers messages. Cette authentification peut être faite soit par un secret partagé préalablement (PSK pre shared KEY, ne pas confondre avec le secret partagé DH), soit par une signature numérique RSA, soit par des méthodes de chiffrement à clé publique qui sont ne sont pas détaillée dans ce document, voire.

http://xml.resource.org/public/rfc/html/rfc2409.html

Freeswan ne permet que l'authentification par secret partagé PSK et par signature numérique RSAsig.

16.3 Analyse de différent mécanisme d'échange des clés pour IPsec

16.3.1 Configuration Host to Lan avec PSK

L'authentification par secret partagé nécessite qu'une information soit préalablement connu des deux interlocuteurs (figure 38). L'initiateur de la connexion transmet le résultat d'une fonction de hachage , définie dans l'échange ISAKMP 1, utilisant le secret partagé.
L'interlocuteur vérifiera l'identité en effectuant la même opération de son côté.

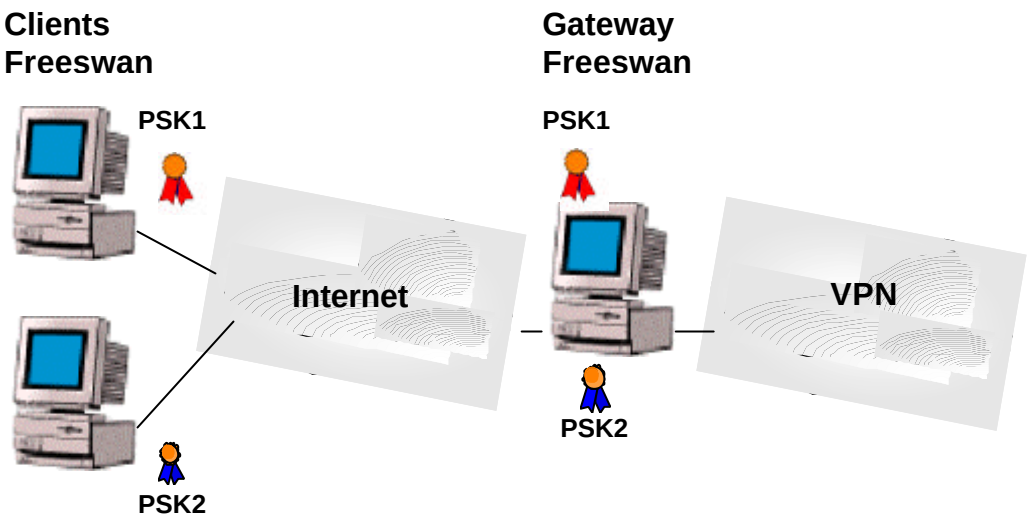


Figure 38 *Connaissance préalable d'un PSK*

Le fichier de configuration du Gateway VPN contenu dans le fichier `ipsec.conf` et défini comme l'extrait de la configuration 22.

```
conn vpn
      left=0.0.0.0
      leftsubnet=
      leftnexthop=
      right=10.192.73.50
      rightnexthop=
      rightnexthop=10.192.72.1
      auto=add
      authby=secret
```

Configuration 22 *Configuration GW pour PSK*

Le secret partagé doit être introduit dans le fichier `ipsec.secrets`

Ce mode d'authentification n'est pas satisfaisant et ne peut pas être utilisé pour deux raisons :

- Le secret partagé PSK doit être échangé de manière discrète, ce qui représente un grave problème du point de vue de la sécurité.
- Le Gateway doit partager un secret avec chaque client. La complexité du système n'est pas concevable pour un grand nombre de client (figure 38)

16.3.2 Authentification par signature RSA

L'authentification par signature numérique utilise la clé privée RSA de l'initiateur pour signer le résultat d'une fonction de hachage.
La signature sera vérifiée à la réception en utilisant la clé publique RSA.

La fonction suivant permet de créer un fichier contenant la clé privée et la clé publique RSA

```
Ipsec rsasigkey –verbose 512 >  /etc/ipsec.secrets
```

La fonction donne un résultat qui a l'allure suivante (configuration 23)

```
: RSA {
      #clé publique RSA, doit être copiée dans le fichier ipsec.conf
      #pubkey=0x012928378bbaksha9a……

      Modulus : 0xcc34js4l9…
      PublicExponent : 0x03

      #clé privée
      PrivateExponent : 0x08847387ce……
      Prime1 : 0xfkeu039j…..
      Prime2 : 0xd9a4f56b4546…
      Exponent1 : 0xa04914fgkdsj059….
      Exponent2 : 0x9129ccfa34….
      Coefficient : 0x75dfac3556…
}
```

Configuration 23 *Clé RSA*

Il s'agit ensuite d'extraire la clé publique dans le fichier `ipsec.secrets` et de l'introduire dans le fichier de configuration `ipsec.conf`

Dans le paramètre `right/leftrsasigkey`

L'exemple suivant définit une configuration d'un Gateway basée sur la signature numérique RSA. (configuration 24)

```
Conn vpn

      Left=10.192.73.60
      Leftid=@toto.ch
      Leftrsasigkey=0x012928378bbaksha9a……

      Right=10.192.73.50
      Rightsubnet=192.168.0.0/24
      Rightnexthop=10.192.72.1
      Rightid=@titi.ch
      Rightrsasigkey=0x03836dhjed8eje8djed94j…

      Auto=add
      #authentification par signature RSA
      Authby=rsasig
```

Configuration 24 *Configuration GW pour RSAsig*

Cette solution d'authentification par signature RSA résout le problème de sécurité posé par le secret partagé. Toutefois elle pose un grave problème, les clés publiques RSA doivent être connues à l'avance et stockées localement avant la connexion (Figure 39).

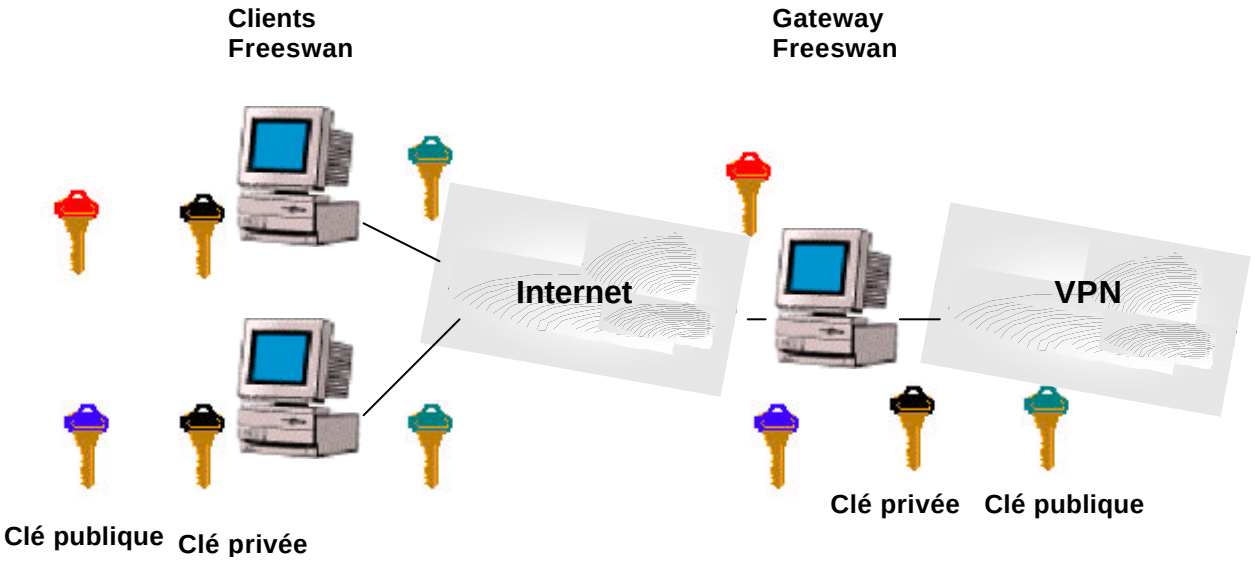


Figure 39 *Connaissance préalable des clés publique RSA*

Les clés publiques pourraient être obtenues dans un annuaire LDAP, mais cette solution reste lourde car elle ne permet pas de se connecter sans manipulation préalable de part et autre.

De plus, l'échange de clés publiques est sujet à l'attaque du « men in the middle ». Il est donc absolument nécessaire d'utiliser l'authentification apportée par les certificats numériques dans tout échange de clé RSA.

16.3.3 Authentification par signature RSA et certificat numérique

Freeswan, comme il a été dit précédemment, ne reconnaît pas les certificats numériques. Il est donc nécessaire de patcher ce logiciel pour permettre l'utilisation des certificats x509. Le patch est contenu dans le paquetage /x509patch-0.9.3-freeswan-1.91

La documentation pour réaliser cette mise à jour est largement détaillée dans le document

http://www.strongsec.com/freeswan/install.htm

Les nouvelles fonctionnalités offertes par ce patch se basent sur les outils cryptographiques de OpenSSL, ce paquetage doit donc nécessairement être installé.

Les clients, comme le Gateway doivent être en possession d 'un certificat numérique. Ces marques d'identité sont obtenues auprès de la PKI OpenCA. Les clients comme la Gateway suivent la procédure conventionnelle pour obtenir un certificat numérique exportable au format PKCS#12. (Laboratoire PKI)

Le paquetage x509patch-0.9.3-freeswan-1.91 met à disposition plusieurs outils permettant d'extraire différents champs d'information contenus dans les certificats numériques.

- Extraction de la clé privée contenue dans les certificats. PKCS#12
- Extraction de la clé publique et de l'identificateur des certificats PEM

L'extraction de la clé privée ne peut être effectuée que par le propriétaire du certificat, étant donné que le certificat PKCS#12 est chiffré par un mot de passe.

La fonction permettant d'extraire cette donnée a été modifiée par C.Tettamanti afin d'introduire automatiquement la clé privée RSA dans le fichier Ipsec.secrets.
La fonction en question est disponible dans le répertoire /x509patch-0.9.3-freeswan-1.91/fswcert

Pour patcher fswcert, copier le patch gupofswcert dans le répertorie /x509patch-0.9.3-freeswan-1.91/fswcert.
Et appliquer le patch

```
Patch fswcert.c gufpofswcert
```

Il est bien évidemment nécessaire de recompiler le code source.

```
Make
```

La commande suivante permet alors d'extraire la clé privée du certificat et de l'introduire directement dans le fichier ipsec.secrets.

```
Fswcert –k –type pkcs12 –p [password} > Ipsec.secrets
```

Le certificat doit être ensuite transmis aux interlocuteurs. Soit le certificat est transmis directement par le propriétaire, soit les correspondant récupèrent les certificats dans l'annuaire LDAP de la PKI.

Dans tous les cas, les certificats doivent être échangés au format PEM, le certificat ne doit bien évidemment plus contenir de clé privée.

La commande suivante permet de convertir un certificat au format PKCS#12 en certificat publiable au format PEM, sans clé privée.

`Openssl pkcs12 –clcerts –nokeys –in {cert.p12} –out {cert.pem}`

Le certificat peut ensuite être transmis à tous les correspondants.

Les certificats échangé doivent être conservés localement dans le répertoire/`etc/ipsec.d/`

Le fichier de configuration `ipsec.conf` peut être nettement simplifié (configuration 25)

```
Conn vpn1

      Authby=rsasig
      Left=0.0.0.0
      Leftcert=client1.pem

      Right=10.192.73.50
      Rightsubnet=192.168.0.0/16
      Rightnexthop=10.192.72.1
      Rightcert=gw.pem
      Auto=add

Conn vpn2

      Authby=rsasig
      Left=0.0.0.0
      Leftcert=client2.pem

      Right=10.192.73.50
      Rightsubnet=192.168.0.0/16
      Rightnexthop=10.192.72.1
      Rightcert=gw.pem
      Auto=add
```

Configuration 25 *Configuration GW pour deux clients x509*

Les paramètres `right/leftid` et `right/`leftrsasigkey ont été remplacés par `leftcert, rightcert`, c'est-à-dire que l'extraction de la clé publique et de l'identité se fera de manière automatique en utilisant les certificats stockés localement.

Si cette méthode d'authentification élimine le risque d'attaque du « men in the middle » lors de l'échange des certificats, elle nécessite toujours un échange préalable des certificats avant toute tentative de connexion. Les certificats obtenus doivent être stockés localement par les tiers (Figure 40).

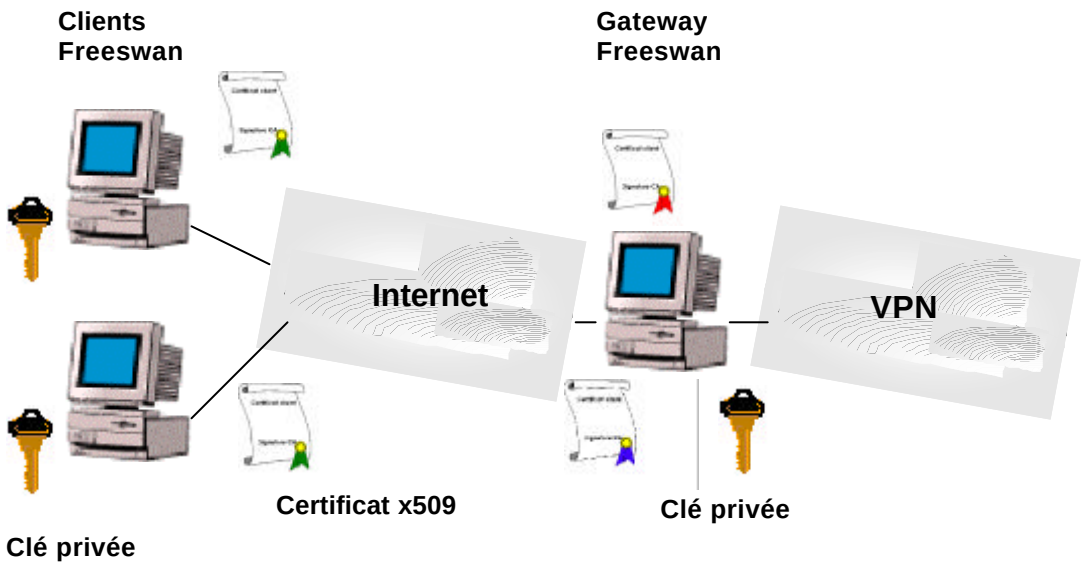


Figure 40 *Connaissance préalable des certificats*

Cette solution n'est pas satisfaisante, car elle requiert une connaissance préalable des tiers et une configuration spécifique pour chaque client Elle ne peut pas être mise en œuvre dans un environnement présentant un grand nombre de client.

16.3.4 Authentification par échange de certificat dans un bloc ISAKMP

Le protocole ISAKMP permet dans sa phase d'authentification, d'échanger des certificats numériques en ligne. Cette possibilité résous tous les problèmes d'échange de certificat précédent. (message 4 Figure 41)

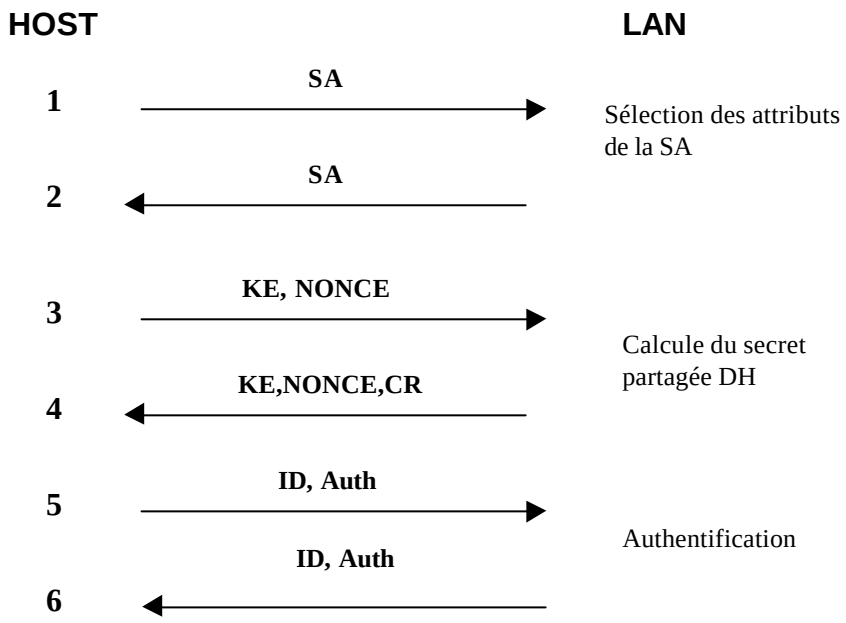


Figure 41 *Echange des certificats*

L'authentification est toujours effectuée en utilisant le principe de la signature RSA, mais le block `CR` utilisé dans le message no4 (Figure 41) indique que le Gateway ne possède pas la clé publique du client. Le client doit transmettre sa clé publique à l'aide d'un certificat dans le bloc `Auth` du message 5.

Cette politique d'authentification est nettement plus souple car elle ne nécessite pas de connaissance préalable des clients. La configuration du Gateway est simplifiée.(configuration 26)

```
Conn vpn
      Authby=rsasig
      Left=%any
      Leftid=%cert
      Leftrsasig=%cert

      Right=10.192.73.50
      Rightsubnet=192.168.0.0/16
      Rightnexthop=10.192.72.1
      Rightrsasig=%cert
      Rightid= « @/Email=pgachet@eivd.ch/CN=GWvpn/OU=TcomCA
      Developper/C=ch »
      Auto=add
```

Configuration 26 *Configuration Gw pour un échange de certificast en ligne*

Le champ `%cert` dans le paramètre `leftid` et `leftrsasig` stipule que ces informations seront fournies ultérieurement par échange de certificats numériques.

Le paramètre `Rightid` contient l'identité du Gateway. Le formalisme permet d'utiliser le `DN` du certificat comme moyen d'identification, mais d'autres formules d'identités peuvent être utilisées comme `FQDN`, ou le formalisme der_`ASN1.`

Pour extraire le `DN` d'un certificat, la commande `OpenSSL` suivante peut être utilisée.

```
Openssl x509 –in {cert.pem} –noout –subject
```

Le Gateway identifiera le client à l'aide du champ transmis dans le paramètre `right/leftid`, il est donc absolument nécessaire que le « DN » transmis corresponde au « DN » contenu dans le certificat, malheureusement tous les attributs définit par x501 ne sont pas supportées par le patch.

Voire http://www.strongsec.com/freeswan/install.htm

16.4 Contrôle des certificats

Comme dans toute connexion utilisant des certificats numériques, une procédure stricte doit être suivie pour contrôler le certificat.

- Contrôle de la date de validité du certificat
- Contrôle de l'intégrité du certificat, en vérifiant la signature de la CA
- Contrôle de la liste de révocation CRL

Pour effectuer ces différents contrôles, les interlocuteurs VPN doivent être en contact régulier avec l'organisme qui a émis les certificats.

16.4.1 Contrôle de la signature de la CA

Pour contrôler l'intégrité du certificat, les tiers doivent posséder le certificat root de la CA. Celui-ci est bien évidement largement distribué, il peut être obtenu auprès de l'interface publique de la PKI ou par l'annuaire LDAP de la RA.

Ce certificat doit être stocké dans le répertoire `/etc/ipsec.d/cacerts.`
La procédure responsable de contrôler automatiquement la validité de tous les certificats échangés dans la phase 1 ISAKMP utilise la clé publique contenue dans le certificat de la CA, la procédure supporte le certificat dans les formats DER et PEM.

16.4.2 Contrôle de la liste de révocation CRL

Avant d'accepter l'identité d'un tiers, il est fortement souhaitable de vérifier que son certificat n'a pas été révoqué. Le VPN à été configurée pour effectuer ce contrôle de manière systématique.

Le scripte `newCRL` a été rédigé de façon à charger la dernière CRL publiée par la PKI.
Il effectue une connexion SSL avec l'interface publique de la PKI, puis charge la CRL dans le répertoire `/etc/ipsec.d/crls.`
Le scripte est lancé automatiquement à date fixe, cette date est synchronisée avec la date de publication de la CRL définie par la PKI.

Si le numéro de série contenu dans le certificat présenté apparaît dans la liste de révocation, la connexion est bien évidemment refusée.

16.5 Configuration en road warrior

Un configuration dite en road warrior, permet à un nombre indéterminé de clients d'établir une connexion Ipsec avec un gateway. Les clients peuvent se connecter depuis différents endroits avec des adresses IP dynamiques.

Cette configuration avec l'authentification par secret partagé PSK est, d'un point de vue de sécurité, irréalisable, car il serait nécessaire que tous les clients partagent le même secret PSK.

En utilisant une configuration par la signature RSA et les certificats numériques stockés localement, le gateway doit disposer d'une configuration différente et spécifique pour chaque client, ce qui est extrêmement lourd.

Seule la configuration par authentification RSA basée sur un échange de certificats en ligne permet de mettre en œuvre une configuration en road warrior.

L'authentification est efficace car elle se base sur RSA et x509. Le gateway n'a pas besoin d'une connaissance préalable des clients, ce qui permet d'utiliser une seule configuration pour

tous les clients. Des configurations spécifiques à chaque client sont générées automatiquement à la réception du certificat, elle se base sur l'identité du client transmis par son DN.

17 Méthode de classification des groupes utilisateurs

17.1 Motivation

Les différents types d'utilisateurs pouvant accéder au VPN doit être assez large, cette gamme d'utilisateurs comprend les étudiants, les professeurs, les assistants et les administrateurs.

Chaque type d'utilisateur dispose de privilèges d'accès différents sur les données contenues dans le réseau interne de l'école.
Ainsi les privilèges accordés à un étudiant peuvent se limiter à l'utilisation du serveur de mail de l'école, de l'accès à un répertoire de partage de fichier. Alors qu'un professeur pourrait bénéficier de l'utilisation de services plus importants comme un accès à un service de téléphonie voIP interne à l'école.

Bien que les services et les privilèges ne soient pas à l'heure actuelle scrupuleusement définit, la problématique du contrôle de ces privilèges peut malgré tout être étudiée avec soin.

L'utilisateur doit en premier lieu être authentifié par la passerelle VPN. Cette authentification est réalisée par l'utilisation de certificat X509 distribuée par la PKI, une fois l'authentification réalisée, le tunnel VPN peut être mis en œuvre pour assuré une confidentialité des messages échangé à l'intérieur du VPN.

Toutefois l'authentification réalisée par les certificats numérique dans le cadre de freeswan, ne permet pas à l'heure actuelle de différencier les différents utilisateurs.
L'authentification permet uniquement de contrôler l'identité d'un client mais ne permet pas de définire les privilèges de ce client lors de la connexion VPN.

Différente solution pour définir des privilèges utilisateur sont étudiée dans cette section, cette étude se base uniquement sur une approche théorique. Aucune ne ses solutions n'a été testée pratiquement.

17.2 Classification par mot de passe et login

La politique la plus populaire d'accès à des services se fait traditionnellement par l'utilisation d'un mot de passe et d'un login. Ainsi le login de l'utilisateur porte en lui tous les privilèges accordés à son propriétaire. Cette politique communément déployée dans l'établissement de session dans un environnement partagé présente quelques graves lacunes de sécurité.

Bien des systèmes d'exploitation regroupent tous les mots de passe et les login des utilisateurs dans un emplacement connu, ainsi un utilisateur avertis pourrait s'approprier ses fichiers et disposer d'informations pertinentes sur tous les utilisateurs.

Bien évidemment, les mots de passe ne sont pas stockés de manière lisible et utilisable immédiatement. Une empreinte par fonction de hachage est souvent utilisée pour les stocker, mais un utilisateur futé peut tout à fait tirer des informations des ces empreintes en comparant leur contenu crypté.
Par exemple, il peut parfois déterminer le nombre de caractères du mot de passe et ainsi, par comparaison, disposer de suffisamment d'informations pour envisager casser le mot de passe par force brute.

Voir http://www.hsc.fr/ressources/presentations/mdp/mdp.htm

De nombreux logiciels très efficaces sont disponibles sur le marché, ils permettent de casser très rapidement les mots de passe généralement triviaux des utilisateurs.

Une solution d'authentification par mot de passe dans le monde Linux utilise le standard PAM (Pluggable Authentification Module). Cette méthode d'authentification permet de gérer différentes alternatives d'authentification comme `login`, `passwd`, `rlogin`, `telnet`, `ftp` de manière simple et souple.

http://www.sun.com/software/solaris/pam/pam.external.pdf;$sessionid$GCZTZ4BXKW4XJAMTA1LU4GQ

http://okki666.free.fr/newbie/linux109.html

Étant donné que l'utilisateur a dû s'authentifier à la passerelle VPN à l'aide d'un certificat numérique, politique d'authentification plus efficace, il est tout à fait raisonnable d'imaginer utiliser ce certificat numérique pour différentier le groupe dont fait partie l'utilisateur.

17.3 Définition des extensions x509v3

La version 3 du standard x509 permet d'affiner la politique de sécurité du certificat en spécifiant différentes extension propres au certificat et à ses composants. Avant de pouvoir envisager une quelconque utilisation de ces extensions pour la définition des groupes VPN, il est nécessaire de définir de façon précise la nature et l'effet de ces extensions.

La norme v3 définit différents groupes d'extension.

- Informations sur les clés
- Informations sur l'utilisation du certificat
- Attributs des utilisateurs et des CA
- Contraintes sur la co-certification

Informations sur les clés

Ce groupe d'extension donne des informations supplémentaires sur l'utilisation des clés, aussi bien les clés du client que les clés de la CA.

Ce groupe est composé de quatre champs

Authority Key Identifier

Ce champs définit de façon unique la paire de clés utilisée par la CA pour signer le certificat. Cette information peut être utile à l'application pour faciliter la procédure de vérification de signature du certificat, dans le cas où la CA aurait utilisé plusieurs paires de clés depuis sa mise en fonction.

Subject Key Identier

Parallèlement à la problématique de l'existence de plusieurs clés pour la CA, l'utilisateur peut également disposer d'un historique de clés qui ne sont plus nécessairement utilisées. Ce champ définit de manière univoque la clé privée associée à la clé publique contenue dans le certificat.

Key usage

Ce champ donne des informations sur l'utilisation qui doit être faite de la clé.
Le champ Key usage peut avoir les valeurs suivantes :

- Non-repudiation
- Certificate signing
- CRL signing
- Digital signature
- Data signature
- Symetric key encryption for key transfert
- Diffie-Hellman key agreement

Le champ key usage peut porter une coloration dite «critique ». C'est-à-dire que la clé ne peut être utilisée que dans le but spécifié, et toute autre utilisation serait contraire à la politique de sécurité définie par la CA et devrait être refusée par les applications. Par contre si ce champs est défini comme « non critique », son but est uniquement indicatif.

Private Key Usage Period

L'utilisation de la clé privée, comme le certificat contenant la clé publique a une durée de vie limitée, la date d'expiration de la clé privée est définie dans ce champ. Il a été mis en évidence dans le tutorial sécurité et PKI, ? ? ? ?, que les clés pouvaient être divisées en deux catégories si l'opération de key recovery était exploitée, les clés utilisées pour signer des documents, et des clés utilisées pour chiffrer des documents. Ce champ ne peut s'appliquer qu'aux clés utilisées pour la signature, car les clés de chiffrement ne doivent jamais expirer.

Information sur l'utilisation du certificat

Ce groupe d'extension contient deux champs qui permettent de définir de manière standardisée l'utilisation propre du certificat. Ces champs sont spécifiés par la PKI suivant la politique de certificat définie de manière formelle dans le document CP (Certificate Policy).

Certificate Policies

Ce champ peut donner plusieurs informations standardisées sur la politique de certification du certificat, il contient l 'OID de la politique de certification utilisée sur ce certificat, il s'agit d'une série de nombres séparés par un point, qui représente de manière standard et universelle les possibilités d'utilisation du certificat.

Ce champ peut contenir plusieurs OID, c'est-à-dire que l'utilisation du certificat est soumis à l'ensemble des contraintes imposées par les différentes politiques, il est dans un tel cas nécessaire que ces différentes politiques ne soient pas incompatibles entre elles.

Comme pour les champs concernant l'utilisation des clés, ce champ peut être stipulé critique, dans tel cas les application doivent respecter scrupuleusement la ou les politiques de certificat définie dans ce champ .

Policy Mapping

Ce champ ne concerne que les certificats des CA elles même, plus particulièrement les CA co-certifiées, c'est-à-dire, le certificat d'une CA qui a signé le certificat d'une autre CA. (Voir Sécurité et PKI 6.6.2).

Ce champ permet d'associer la politique de certification de la CA à la politique de certification de la CA qui a signé son certificat, ce champ rend possible une interopérabilité en manière de politique de certification indispensable dans un modèle Peer to Peer.

Attributs des utilisateurs et des CA

Ce groupe d'extension contient trois champs qui donnent des informations plus précises sur l'identité des utilisateurs des certificats. Il est composé de trois champs.

- Champ sur le nom des utilisateurs.
- Champ sur la co-certification.
- Champ sur la politique de certification.

Subject Alternative Name

Ce champ contient un ou plusieurs noms supplémentaires pour identifier le propriétaire du certificat. Les applications de messageries, comme les application Web, peuvent exploiter ces informations de manière plus fine que la seule information contenue dans le DN du propriétaire.

Ainsi les noms autorisés dans cette extension sont :
1.Une adresse mail
2.Un nom de domaine
3.Une adresse IP
4.Une nom supplémentaire
5.Une URL
6.Un nom défini par une OID

Issuer Alternative Name

Ce champ permet d'avoir plus d'informations sur la CA qui a émis le certificat, les noms autorisés sont les mêmes que pour le champ précédent.

Contrainte sur la co-certification

La possibilité de mettre en œuvre une PKI intéropérable avec une autre PKI en utilisant la procédure de co-certification nécessite une définition strict des niveaux de contrôle et de confiance envers d'autre CA. Ces contraintes sont stipulées dans ce groupe d'extension.

Basic contraints

Ce champ indique si l'utilisateur est un utilisateur final ou si il peut être un CA. Dans le cas où l'utilisateur est un CA, le certificat est forcement co-signé. Ce champ contient dans ce cas « une distance de certification ». Cette information spécifie jusqu'où doit remonter une application qui désire vérifier un certificat en consultant sa CRL, l'information définit également l'étendue de la chaîne de confiance. Si la distance est à un, les utilisateurs ne peuvent que vérifier les certificats émis directement par la CA.

Name contraints

Ce champ, également valide uniquement dans le cas d'une co-certification, permet aux administrateurs de limiter les domaines de confiance accordée dans le domaine de certificat délivré par la CA co-certifiée. Par exemple, la PKI mise en œuvre a délivré des certificats à des individu de confiance (ami), ces ami sont représenté dans l'annuaire LDAP par l'entrée suivante. Dn : cn= (ami), o=tcom, c=ch

Une autre PKI a également délivré des certificats à des personnes de confiance, mais aussi à des visiteurs.

```
Dn : cn =(visiteur), o=autrepki, c=ch
Dn : cn=(ami),o= autrepki, c=ch
```

Les deux PKI sont interopérables par co-certification, mais aucune confiance n'est donnée au visiteur. Le co-certificat ne sera donc valide que pour la branche

```
Dn : cn=(ami),o=autrepki,c=ch
```

Policy constraints

Ce champ permet de stipuler la politique de certification acceptable pour les certificats dépendants du co-certificat, ce champ n'a de sens que pour un système co-certifié.

17.4 Définition des groupes d'utilisateurs par les extension V3

Parmis les différents groupes d'extension décrit ci-dessus, seul le groupe « Attributs des utilisateurs et des Ca » permet de définir de manière supplémentaire le groupe d'utilisateur auquel appartient l'utilisateur du certificat par exemple étudiant, professeur.

Le champ « nom supplémentaire » ou « nom définit par une OID » peut contenir cette information.

Lorsque l'on définira une application particulière, par exemple le service de messagerie, il est dès lors possible d'effectuer un contrôle sur un champ d'une extension particulière, dans notre cas, le nom défini par une OID. La norme v3 permettant de stipuler de manière stricte quels champs de l'extension doivent être pris en compte.

Une limitation importante apparaît dans cette politique de classement, il est possible que l'utilisateur change de profil, par exemple q'un étudiant devient assistant ou professeur, la spécification du groupe d'utilisateurs étant introduite de manière statique dans le certificat empêche toute modification du profil de l'utilisateurs. L'utilisateur doit donc effectuer par l'intermédiaire de la RA concernée une demande de révocation de certificat, puis la CA doit signer cette demande et publier la CRL, par la suite le client doit informer la RA de la modification de son profile utilisateur, l'administrateur de la RA aura la tâche de générer une demande de certificat d'utilisateur avec le nouveau profil, celle ci une fois transmise au CA et signée pourra être retournée à l'utilisateur qui bénéficiera de nouveaux privilèges.

Cette procédure est à l'évidence assez lourde et se prête difficilement à des modifications de profil utilisateur généralisé. Pour cette raison, la politique PKI déconseille d'introduire des informations sur les privilèges des utilisateurs dans les extensions, les extensions ont pour but de définir la politique d'utilisation des certificats et non pas les privilèges propres à l'utilisateur. Par exemple, l'extension peut limiter l'utilisation du certificat à la signature numérique, mais pas à un cryptage de courrier électronique.

Pratiquement, le gateway VPN n'est pas en mesure d'interpréter les extensions des certificats x509. Il s'agirait de modifier considérablement les sources de freeswan pour permettre une telle prise en charge.

17.5 Classement par signature de la CA

La structure d'une PKI fonctionne sur le principe d'une chaîne de confiance hiérarchique, ainsi la CA peut déléguer le travail de signature à des CA subordonnées, les certificats de ces CA sont signées par la CA root mais toutes les CA subordonnées pourront signer elles-mêmes les requêtes de certificat qui leur parviennent.

La politique de classement des utilisateurs suivant la CA qui a signé le certificat permet de bénéficier de cette hiérarchie.(Figure 42)

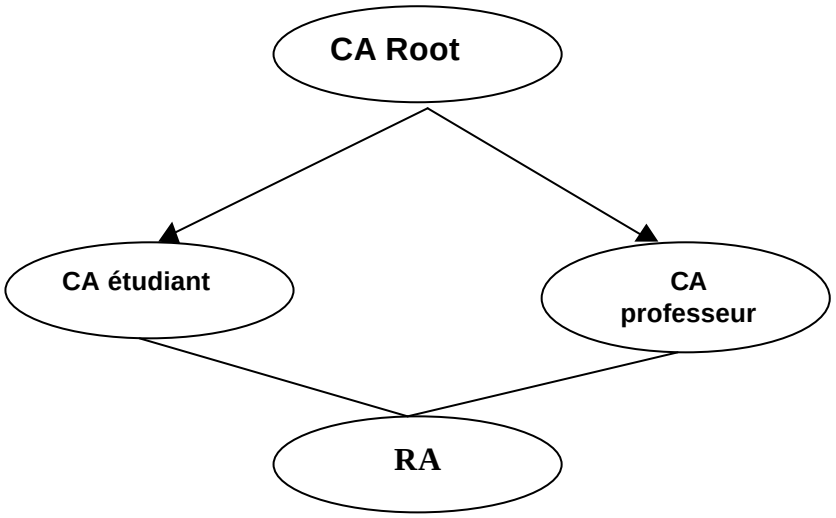


Figure 42 *Classification par signature CA*

Ce schéma hiérarchique définit deux CA subordonnées l'une pour les étudiants et une pour les professeurs. Lorsque l'utilisateur désire obtenir un certificat, l'administrateur de la RA choisira vers quelle CA envoyer la requête de certificat suivant le profil de l'utilisateur.

Le profil de l'utilisateur sera donc intrinsèquement lié à l'autorité qui à signé le certificat, les applications devront analyser la signature du certificat présenté pour définir si l'utilisateur à les privilèges suffisant pour accéder à l'application.

Cette politique présente le même désavantage que précédemment, c'est à dire que les privilèges utilisateur sont maintenu de manière statique à l'intérieur même du certificat.

Toutefois la procédure de modification des profils utilisateur est quelque peut simplifiée, l'administrateur de la RA peut simultanément envoyer une demande de certificat à un CA et envoyer une demande de révocation de certificat à l'ancienne CA car ces deux autorité travaillent en parallèle.

Cette politique est très séduisante car elle permettrait de définir deux zones VPN (Figure 43), chaque zone ne serait accessible que par des utilisateurs dont le certificat a été signé par une CA spécifique.

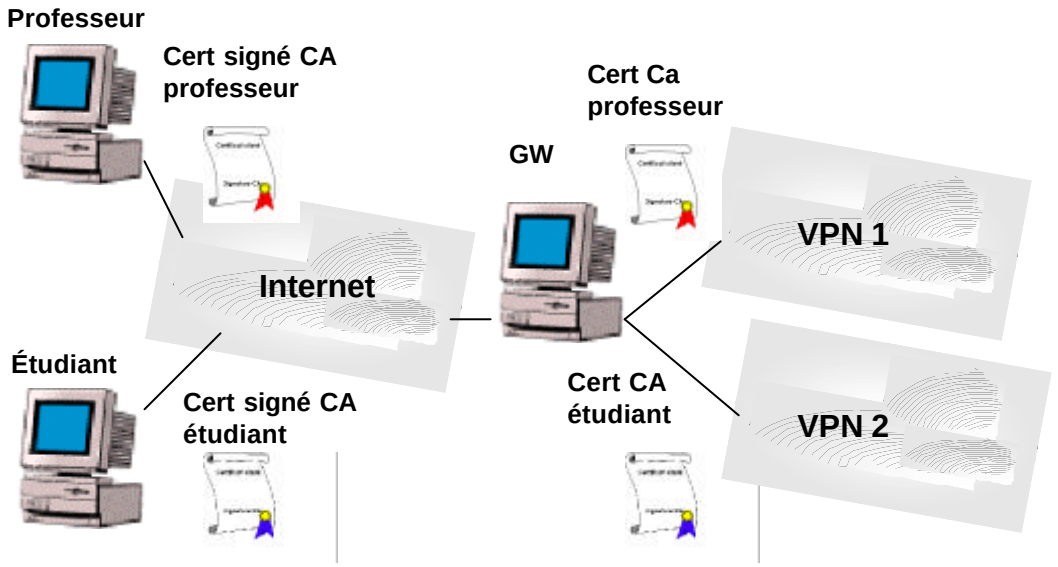


Figure 43 *VPN basé sur la signature de la CA*

Pour le moment, le patch de freeswan ne permet pas d'utiliser différentes autorité de certification. Pour mettre en œuvre une telle configuration (Figure 43), il serait nécessaire de modifier freeswan pour permettre la reconnaissance de plusieurs signature de CA.

17.6 Classement d'après le DN du certificat

La Pki OpenCA permet d'apporter une certaine coloration dans les certificats générés. Les clients PKI sont divisés en trois groupes d'utilisateurs.

Ces groupes sont définis par un attribut OU dans le DN du certificat, par exemple un utilisateur normal appartiendra au groupe OU=tcomCA User, alors qu'un administrateur aura le privilège d'appartenir au groupe OU=tcomCa Developper.

Cette catégorisation d'utilisateurs avait déjà été exploitée lors de la connexion au serveur Apache de la RA (voire13.5.6)

Le serveur Apache mettait à disposition des outils qui permettaient de refuser l'accès au serveur à tous les clients ne présentant pas un certificat du groupe « `tcomCa Developper` ».

Ce niveau de granularité n'est pas possible avec le Gateway freeswan, celui ci ne permet pas de définir différents niveaux de contrôle sur les certificats. Pour permettre une telle solution, il est nécessaire d'effectuer des modifications dans les sources des fichiers freeswan. Cette solution a été envisagée et les développeurs de patch ont été contactés pour information.
Ils déconseillent catégoriquement toute tentative de manipulation sur le champ DN du certificat.

La classification des utilisateurs au niveau VPN (Ipsec) n'est pas la seule solution, il est possible de déléguer cette sélection aux applications de couche supérieur.

17.7 Contrôle d'accès centralisé

L'importance des échanges d'informations et des différents serveurs d'application partagés à l'intérieur des entreprises ont mis en évidence la problématique du contrôle d'accès. Une solution qui tend à devenir généralisée, consiste à déléguer le contrôle d'accès de manière centralisée et de définir une politique d'accès standardisée à tous les services. Cette politique ne se base plus directement sur les certificats X509.

L'association ECMA (european computers manufacturers association) a défini un standard de protocole de sécurité pour des applications ouvertes et hétérogènes. Le projet se nomme SESAME(Secure European Systeme in A Multi-vendor Environment), il a pour but de résoudre les questions d'accès aux ressources partagées pour définir une politique générale de sécurité.

L'authentification dans le cadre de SESAME, se base sur Kerberos mais ajoute à ce système des fonctions supplémentaires, par exemple des contrôles de login exploitant aussi bien la signature numérique que l'utilisation de certificat délivré par une PKI.

Son objectif premier était d'étendre la notion d'authentification au contrôle d'accès basé sur le rôle des utilisateurs. RBAC (Role Based Access Control). Dans les systèmes basés sur RBAC, le droit d'accès n'est pas défini par le nom des utilisateurs, mais bien par le rôle des utilisateurs, le rôle définit la fonction de l'utilisateur, par exemple administrateur, étudiant, professeur…

L'avantage d'une politique d'accès basée sur les rôles réside dans la souplesse sans pareille qui incombe à la gestion du système. L'administrateur du système n'a plus besoin de définir une politique d'accès personnalisée pour chaque utilisateur, mais plus simplement de mettre à jour une liste des rôles et des privilèges associés ACL (Access Control List).

De cette manière, l'ajout d'un nouvel utilisateur devient une procédure tout à fait simple, car l'identité propre de cet utilisateur n'est pas significative, seul son rôle doit être défini.

Le contrôle d'accès peut être géré de façon centralisée dans une base de données administrée par un responsable de la sécurité. Cette bases de donnée porte le nom de PAS (Privilege Attribute Server).

Tous les profils des utilisateurs et leur rôles sont définis dans cette base, ainsi l'utilisateur voulant se connecter à une certaine application devra en premier lieu s'authentifier au PAS, cette authentification se fait d'après la politique Kerberos, c'est-à-dire par ticket de service(voir tutorial sécurité et PKI 5.2.3), le PAS lui fournira un PAC (Privilege Attribute Certificate), ce PAC sera par la suite présenté aux applications qui le contrôleront.

Le standard ECMA –219 PAC définit deux catégorie de PAC

- PAC Non-delégable
- PAC délégable

PAC Non-delégable

Cette catégorie de certificats est utilisée par un utilisateur dont l'identité est définie, le certificat est protégé par un PPID (Primary Principal Identifier).
Les informations contenues dans ce PAC donnent des indications sur la session envisagée.

PAC délégable

Cette catégorie de PAC permet de confier temporairement des privilèges et des droits d'accès à d'autres utilisateurs. Bien évidemment ce procédé ne doit pas être une faille dans la sécurité du système, entendu par là qu'un utilisateur mal intentionné ne doit pas pouvoir s'approprier les privilèges véhiculés par le PAC.

Pour renforcer la sécurité, les PAC délégable contiennent une « Check Value (CV) », cette information est calculée en utilisant une fonction à sens unique sur un nombre aléatoire PV (Protection Value) générée par le PAS.

L'utilisateur qui obtient un PAC délégable, reçoit également du PAS un PV chiffré à l'aide de la clé de session en cours. Lors que cet utilisateur transmet son PAC à un autre individu qui partage une session, le propriétaire du PAC transmet également le PV chiffré cette fois à l'aide de la clé de session commune entre les utilisateurs.

L'utilisateur qui reçoit le PAC délégable connaît aussi le PV correspondant au CV contenu dans le PAC, il est donc le seul en mesure d'utiliser le PAC.

Les deux types de PAC ont en commun une validité temporelle limitée, généralement un PAC n'est valide que pendant quelques heures pour limiter au maximum les risques éventuelles. (voir 5.2.4)

Le PAS fournit aux utilisateurs un PAC (Privilege Attribute Certificates) qu'ils présenteront aux applications. Cette politique d'attribution de privilèges ressemble à la politique

d'authentification Kerberos qui utilise des tickets d'accès pour s'authentifier auprès des serveurs, cette similitude n'est pas une coïncidence car les deux technologiques ont été développées par le même fournisseur. Le service PAC est une extension du protocole Kerberos standardisée par ISO.

Une documentation plus précise du service PAC est disponible depuis l'URL suivant.

http://www.esat.kuleuven.ac.be/cosic/sesame/papers/cms97.html

Conclusion

Pour l'instant, aucune solution n'est réalisable sans travail supplémentaire. Pour établir une sélection au niveau VPN, seule la solution basée sur la signature de la CA semble réaliste, mais elle nécessiterait.

- Mettre sur pied plusieurs CA subordonnées (basée sur OpenCA), une pour les professeurs et une pour les étudiants. Les deux Ca devraient être certifiée par la CA root.
- Modifier les sources de freeswan pour rendre possible la reconnaissance de plusieurs CA.

L'autre solution consisterait à déléguer la sélection au niveau application par l'intermédiaire d'un système RBAC comme SESAME, c'est à dire qu'il faudrait mettre sur pied un second système d'authentification, par exemple Kerberos.

Références

[1] implémentation d'une solution Ipsec en opensource
http://freeswan.org
[2] Configuration freeswan
http://www.freeswan.org/freeswan_trees/freeswan-1.9/doc/config.html
[3] intégration des certificat x509 pour freeswan
http://strongsec.com
[4] Public key extensions to SESAME authentification protocols
http://www.esat.kuleuven.ac.be/cosic/sesame/papers/cms97.html
[5] The OAKLEY key Determination Protocol
http://www.twi.ch/~sna/research/VPN/rfc/rfc2412.html
[6] The Internet Key Echange (IKE)
http://www.twi.ch/~sna/research/VPN/rfc/rfc2409.html
[7] The Internet IP Security Domain of Interpretation for ISAKMP
http://www.twi.ch/~sna/research/VPN/rfc/rfc2407.html
[8] Projet SEAME
http://www.ida.liu.se/labs/iislab/courses/ANS/Lesson6/ppframe.htm
[9] How Role Based Access Control is implemented in SESAME
www.esat.kuleuven.ac.be/cosic/sesame/papers/wetice97.pdf
[10] Certificats x509v3
http://sw00d.free.fr/crypto/pages/x509v3.html
[11] Crackage et durcissement des mots de passe
http://www.hsc.fr/ressources/presentations/mdp/mdp.htm

18 Conclusion générale

Ce travail de diplôme aura permis de s'initier aux mécanismes passionnants de la cryptographie ; depuis l'étude des procédures mathématiques jusqu'à l'utilisation pratique des algorithmes cryptographiques dans des cas concrets.
Si la cryptographie est vue comme une arme redoutable par certains états, son utilisation s'intègre malgré tout parfaitement à nos moyens de communication moderne. Ce double aspect n'apporte que plus d'intérêt à son étude.
D'une part l'étude d'une technologie moderne, pas encore intégrée au programme de base de l'école est toujours un atout sur le marché de l'emploi. Deuxièmement, le simple fait d'être autorisé à mettre en pratique des applications aussi critique qu'une PKI on été perçu comme un privilège de taille, permettant d'assouvir sa curiosité sur des technologies apparemment élitistes.

Néanmoins, l'étude d'une technologie en soi n'a aucun sens, si elle ne peut apporter de solutions nouvelles. Et ces solutions ont été apportées par la mise sur pied de la PKI.
La PKI a permis de résoudre de façon élégante et efficace le problème d'authentification dans le cadre du VPN. En résolvant le problème de l'authentification, le problème des utilisateurs nomades s'est résolu de lui-même. Ainsi un grand nombre de clients peuvent se connecter au VPN depuis n'importe quel emplacement, le simple fait de présenter un certificat numérique pendant la phase d'authentification suffit à différencier et à définir une connexion propre à chaque client.

La technologie PKI ayant le vent en proue, de plus en plus d'applications commencent à intégrer le standard x509. Ainsi, la PKI mise en œuvre ne pourrait pas se limiter à collaborer avec le VPN, mais pourrait devenir une entité polyvalente à toutes les applications PKI-ready qui pourraient être développées dans le cadre de l'école.

Toutefois le monde PKI est nettement plus vaste qu'il n'y paraît. Dans ce travail de diplôme, la technologie n'a été qu'entrevue. Seuls les mécanismes indispensables permettant l'intégration avec un VPN ont été développés. La mise en œuvre de la PKI a permis par exemple d'ouvrir une porte sur la technologie LDAP, sans pour autant s'aventurer dans ce monde. De ce fait, de nombreux aspects sont encore à découvrir, soit directement dans la technologie PKI, soit dans les technologies partenaires.
La PKI qui a été mise sur pied, n'a pas pu pour l'instant résoudre le problème des droits d'accès et des privilèges des groupes utilisateurs dans le cadre du VPN. Il ne s'agit pas pour autant d'un manque d'efficacité de la technologie PKI tout entière, mais plutôt d'un manque de maturité des implémentations qui ont été utilisées.
En choisissant de travailler dans un monde ouvert comme LINUX, on a pu bénéficier de manière gratuit des travaux et du savoir faire de divers communautés. Mais on ne peut évidemment pas s'attendre à obtenir la même maturité qu'un produit PKI commercial.
Toutefois l'univers opensouce en plus de fournir des implémentations de manières gratuit, fournit la possibilité de modifier le code à souhait, permettant ainsi d'adapter de façon efficace du code source. Si une fonctionnalité manque, sous LINUX, il est toujours possible de la créer.

Annexe

Annexe A Sigles et acronyme

3DES	Triple-DES Encryption protocol
AH	Authentification Header
ACL	Access Control Lists
ASN.1	Abstract Syntax Notation One
BD	Base de données
BTP	Back Traffic Protection
CA	Certificate Authority
CAM	Code d'authentification de Message
CBC	Cipher Block Chaining
CCTI	Centre de Compétence de HES-SO dans les Technologies de l'Information
CFB	Cipher Feed Back
CP	Certification Practice Statement
CPS	Certificate Practice Statement
CSR	Certificate Signing Request
CRL	Certificate Revocation List
CV	Check Value
DES	Data Encryption Standards
DH	Diffie-Hellman
DIT	Directory Information Tree
DN	Distinguished Name
DSA	Digital Signature Algorithm
DOI	Domain of interpretation
ECB	Elecronic Code Book
ECMA	European Computers Manufactuers Association
ERP	Entreprise Ressource Planning
ESP	Encapsulating Security Payload
GW	Gateway
HSM	Hardware Storage Module
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol over Secure Socket Layer
IAB	Internet Architecture Board
IETF	Internet Engineering Tak Force
IKE	Internet Key Echange
IP	Internet Protocol
IPSec	IP security protocol
ISAKMP	Internet Security association and Key Management Protocol
ISO	International Standardization Organization
IV	Initialization Vector
KRM	Keon Key Recovery Module
LDAP	Lightweight Directory Access Protocol

LDIF	Ldap Data Interchange Format
MAC	Message Authentification Code
MD5	Message Digest 5
NIST	National Institute of Standards and Technology
OCSP	Online Certificate Status Protocol
OID	Object IDentifier
OSI	Open Systems Interconnection
OU	Organization Unit
PAC	Privilege Attribute Certificate
PAS	Privilege Attribute Server
PEM	Privacy Enhanced Mail
PFS	Perfect Forward Secrecy
PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure
PGP	Pretty Good Privacy
PSK	Pre Shared Key
PPID	Primary Principal Identifier
PPTP	Point-to-Point Protocol
PV	Protection Value
RA	Registration Authority
RBAC	Role Based Access Control
RFC	Request For Comments
RSA	Rivest, Shamir,Adleman
SA	Security Association
SAD	Security Assosiation Database
SESAME	Secure European System in A Multi-vendor Environment
SHA	Secure Hash Algorithm
SKEME	A Versatile Secure Key Exchange Mechanism for Internet
SKIP	Simple Key management for Internets Protocols
S/MIME	Secure Multi-Purpose Internet Mail Extensions
SPD	Security Policy Database
SPKI	Simple Public Key Infrastructure
SPI	Security Parameter Index
SSH	Secure Shell
SSL	Secure Socket Layer
TCP	Transport Layer Protocol
TGS	Ticket Granting Service
TGT	Ticket Granting Ticket
TLS	Transport Layer Security
TTL	Time To Live
UDP	User Datagram Protocol
VPN	Virtual Private Network